# Ora anche su disco

MS-DOS & GW-BASIC EMULATOR

TRASFORMA IL TUO 64 IN UN PC IBM

*"MS-DOS & GW-BASIC emulator" è anche su disco. Per quanti hanno acquistato la versione su cassetta ed inviano la relativa prova d'acquisto, il dischetto è disponibile a lire 15.000 (+ lire 3.000 per spese di spedizione). Non occorre inviare la cassetta nè tantomeno il manualetto di istruzioni. Chi non è in possesso della cassetta può richiedere il disco ed il manualetto al prezzo normale di lire 25.000 (+ lire 3.000 per spese di spedizione).*

*Per una veloce evasione dell'ordine inviate un assegno bancario o circolare non trasferibile all'ordine della "Systems Editoriale" (V.le Famagosta, 75 - 20142 Milano).*

***Sempre un passo avanti.***

# Sommario



## INSERTO



## RUBRICHE



**Direttore:** Alessandro de Simone - **Caporedattore:** Michele Maggi
**Redazione/collaboratori:** Paolo Agostini, Claudio Baiocchi, Carlo e Lorenzo Barazzetta, Simone Bettola, Luigi Callegari, Sergio Camici, Sandro Certi, Umberto Colapicchioni, Maurizio Dell'Abate, Valerio Ferri, Giancarlo Mariani, Roberto Marigo, Clizio Merli, Marco Miotti, Roberto Morassi, Antonio Pastorelli, Carla Rampi, Marco Saetta, Fabio Sorgato, Danilo Toma, Giovanni Valli.
**Segretaria di redazione:** Maura Ceccaroli **Grafica:** Arturo Ciaglia, Gabriella Galbusera
**Copertina:** Immagine da Linea Intima 4/86 per gentile concessione della Miti-Bayer e dell'agenzia Donada
**Direzione, redazione, pubblicità:** v.le Famagosta, 75 - 20142 Milano - Tel. 02/8467348
**Pubblicità:** Milano: Leandro Nencioni (direttore vendite), Guido Agosti, Giorgio Ruffoni, Claudio Tidone - v.le Famagosta, 75 - 20142 Milano - Tel. 02/8467348
• Emilia Romagna: Spazio E - P.zza Roosevelt, 4 - 40123 Bologna - Tel. 051/236979
• Toscana, Marche, Umbria: Mercurio srl - via Rodari, 9 - San Giovanni Valdarno (Ar) - Tel. 055/947444
• Lazio, Campania: Spazio Nuovo - via P. Foscari, 70 - 00139 Roma - Tel. 06/8109679
**Segreteria:** Marina Vantini - **Abbonamenti:** Liliana Spina
**Tariffe:** prezzo per copia L. 4.000. Abbonamento annuo (11 fascicoli) L. 40.000. Estero: il doppio. Abbonamento cumulativo alle riviste Computer e Commodore Computer Club L. 80.000. I versamenti vanno indirizzati a: Systems Editoriale Srl mediante assegno bancario o utilizzando il c/c postale n. 37952207
**Composizione:** Systems Editoriale Srl - **Fotolito:** Systems Editoriale Srl
**Stampa:** La Litografica Srl - Busto Arsizio (Va)
**Registrazioni:** Tribunale di Milano n. 370 del 2/10/82 - Direttore Responsabile: Michele Di Pisa
Sped. in abb. post. gr. III - Pubblicità inferiore al 70% - **Distrib.:** MePe - via G. Carcano, 32 - Milano

# Verso un nuovo Boom?

*Sembrerebbe che vi siano le premesse per duplicare il successo di vendite di qualche anno addietro*

di Alessandro de Simone

Mai, come in questi ultimi mesi, siamo stati bombardati da pubblicità relativa a computer, periferiche, accessori, software ed altre diavolerie che, finalmente, hanno convinto gli studi professionali, e gli "uffici" in generale, della necessità di automatizzare numerose procedure.

La IBM ha fatto scalpore annunciando, mesi fa, la nuova serie di computer il cui slogan di vendita è: "Il passato si può copiare, il futuro si deve creare", con chiaro riferimento alla eccessiva (e fastidiosa?...) prolificità di computer fatti ad immagine e somiglianza del "Big Blue", come viene anche chiamata la IBM.

Anche molte altre multinazionali dell'informatica, tra cui l'Olivetti, si sono precipitate a dichiarare la disponibilità di computer della nuova, nuovissima e, magari, futura e futurissima generazione di elaboratori.

Chi è addentro nel settore, però, sa benissimo che la IBM, più che a limitare la diffusione dei cloni PC (che, paradossalmente, hanno aumentato le vendite proprio dal momento dell'annuncio di Big Blue) sembra voglia re-introdursi nel settore dei microcomputer, di cui era indiscussa leader prima che la Digital proponesse un modello niente male.

Nel campo hobbystico, invece, il predominio spetta ancora alla Commodore, e sono pochi coloro che vogliano rischiare di esser smentiti affermando il contrario.

Ma ciò che a noi utenti Commodore interessa di più, ovviamente, non è tanto la politica commerciale di altre aziende del settore (che, comunque, in un modo o nell'altro, influenzano l'intero settore) ma, soprattutto, interessa sapere che cosa bolle in pentola, che cosa verrà proposto e che cosa, insomma, ci dobbiamo aspettare in un prossimo futuro.

Il mercato dell'informatica hobbystica, infatti, è ormai maturo o, se preferite, ha perso la caratteristica della disinformazione e dell'avventura.

Chi, oggi, intende acquistare un computer, si può documentare, prima dell'acquisto, molto di più di chi affrontava la spesa qualche anno fa.

Ne consegue che, nel corso di questi ultimi anni post-boom, le aziende del settore, volenti o nolenti, si sono dovute adeguare, finalmente, alle esigenze dell'utente ormai conscio dei suoi diritti. Ciò, in parole povere, si traduce nella migliore qualità dei servizi che, nel campo dell'informatica, significa soprattutto assistenza post-vendita.

In questa ottica, quindi, nessuna meraviglia deve destare il rinnovato interesse per il settore, per il software e per gli accessori in generale.

E se, per il prossimo Natale, vedrete nuovamente la coda davanti ai negozi di computer, non meravigliatevi: sono i segni della nuova generazione; non tanto di calcolatori, quanto di utenti...

## Uffa!

**☐ Ho digitato il programma "Disassembler" (CCC N.41) ma, dopo il Run, si blocca emettendo il messaggio "Out of data error" in 1190. Perchè mai c'è tale segnalazione dal momento che il listato da me digitato (che allego in output di stampante) è perfettamente eguale a quello da voi pubblicato?**
*(Andrea Benelli - Firenze)*

• Il listato che hai inviato su carta è perfettamente eguale a quello pubblicato tranne che... nell'ultimo dato. Tu hai concluso con "-1" mentre il programma pubblicato termina con "-2". Tale valore è stato intenzionalmente inserito da F.Sorgato (uno dei nostri più validi e stimati collaboratori) proprio per far "capire" al programma quando è il momento di smettere di leggere i Data mediante Read (vedi riga 1210).

Pubblicando questa lettera ne approfitto per ricordare ai lettori che non possiamo prendere in considerazione contestazioni sui listati pubblicati, a meno che i programmi digitati non pervengano su disco per un più efficace, e soprattutto rapido, controllo.

Il 999.999% delle volte, infatti, come appunto in questo caso, gli errori sono dovuti ad imprecisioni commesse da parte dei lettori stessi.

## Istruzioni 1541

**☐ Mi hanno regalato un drive 1541 il cui libretto di istruzioni è in inglese. E' possibile procurarselo in italiano?**
*(Mauro Cianchi - Genova)*
*(Giorgio Parri)*

• La Commodore Italiana ha provveduto, fin dal 1984, ad allegare libretti in italiano (oltre a quello originale in inglese) alla confezione del drive 1541: te lo posso assicurare dal momento che ne posseggo una copia.

E' probabile che, in seguito, abbiano deciso di privare i nuovi acquirenti del comodo manuale, oppure (caso più probabile) che il tuo drive sia stato acquistato da uno dei cosiddetti "canali paralleli" altrettanto attivi di quelli utlizzati per le automobili tedesche (la Commodore ha stabilimenti di produzione in Germania).

Non credo, però, che il manuale, pur se esauriente e chiaro, sia proprio di vitale importanza per un possessore di drive. Non dimenticare che la nostra rivista, molto più spesso di quanto tu non creda, pubblica articoli sul modo di utilizzare al meglio la valida periferica, richiamando, per ovvi motivi di completezza, sia notizie tratte dal manuale, sia procedure inedite.

## Ondeggiamenti

**☐ Perchè quando accendo il drive 1571 lo schermo ondeggia?**
*(Antonio De Giorgi - M.Leuca)*

• Quando si accende un qualsiasi apparecchio elettrico, viene chiesta alla "linea" un brusco incremento di consumo. La linea ha quindi bisogno di adattarsi alla nuova richiesta e durante tale periodo di tempo (in genere intorno al secondo) si ottiene un abbassamento della tensione. I circuiti elettrici del TV risentono delle variazioni e l'effetto che ne risulta consiste in alterazioni dell'immagine.

Se, però, il difetto persiste, la causa non può essere quella accennata dal momento che, lo ripeto, dovrebbe durare pochissimo tempo.

E' più probabile, quindi, che gli ondeggiamenti persistenti siano dovuti ad interferenze elettromagnetiche indotte dall'alimentatore incorporato nel drive sui delicati circuiti del TV.

Prova ad allontanare quanto più è possibile il drive (e l'alimentatore del computer) dal video: il difetto dovrebbe scomparire del tutto.

Lo stesso inconveniente si verifica anche tra due monitor posti troppo vicini tra loro.

## Non aprire quella porta

**☐ Avevate promesso molto tempo fa (addirittura su CCC N.34), che avreste pubblicato quanto prima un articolo su come utilizzare la User Port del C/64. In particolare vorrei collegare direttamente un motorino da 12 Volt. Posso?**
*(Paolo Scalabrini - Campomorone)*

• Noooo! Il motivo principale per cui stiamo tardando la pubblicazione di articoli del genere è dovuto proprio alla eccessiva disinvoltura con cui molti lettori userebbero la porta utente.

Qualcuno invece, subdolamente, ritiene che noi non siamo in grado di affrontare l'argomento e perciò tentenniamo e la tiriamo per le lunghe.

Se per il momento vi accontentate, comunque, possiamo comunicare che l'altra rivista che pubblicavamo nel 1984, dal semplice nome "Commodore" riporta (N.3, pag.18 articolo "C.I.A.") non solo la teoria di funzionamento ma addirittura il circuito stampato da realizzare ed il software per gestirlo.

Ai più pazienti e fiduciosi non possiamo che rinnovare l'invito ad aspettare la presentazione di un progettino di semplice realizzazione ed alla portata di tutte le esperienze (e tasche).

Se proprio volete procurarvi il fascicolo "Commodore" N.3, telefonate al nostro servizio arretrati (02/84.67.34.8) chiedendo della signora Lucia.

## Padre & figlia

**☐ Posseggo da poco un C/64 e, aiutato da mia figlia Ilaria di otto anni, ho migliorato un vostro programma di cui invio copia su carta. Sarei orgoglioso se poteste pubblicarlo...**
*(G.Zucchetta - Genova Voltri)*

• Purtroppo, come ho già precisato altre volte, non mi è possibile digitare listati che pervengano solo su carta; del resto, anche lo avessi ricevuto su

supporto magnetico, dubito che lo avrei pubblicato: un programma, infatti, pur se ampliato in modo notevole e interessante (come è il caso nostro), può essere riproposto ai lettori solo se presenta cambiamenti sostanziali rispetto alla versione già nota.

Complimenti, comunque, per l'impegno mostrato (ma davvero è un principiante?) e un bacione alla simpatica Ilaria che "sopporta" pazientemente il suo bravo papà...

## On error goto

☐ **Vorrei maggiori chiarimenti circa il comando "On error Goto" del vostro simulatore Gw-Basic.**
*(Giuseppe Ornaghi - Chieti)*

• Supponiamo di digitare il seguente programma, naturalmente dopo aver caricato ed attivato l'emulatore Gw-Basic:

```
110 input"dividendo";x
120 input"divisore";y
130 z=x/y: rem errore?
140 print"il rapporto è"x/y
150 print: goto 110
```

Il listato richiede due valori e ne calcola il rapporto. Dovrebbe esser noto che non è possibile eseguire la divisione per zero; se si tentasse egualmente (esempio "Dividendo: 1; Divisore: 0") si otterrebbe il messaggio "Division by zero error in 130". Per evitare che il programma si interrompa, nonostante si verifichi un errore, aggiungi le righe fino a pervenire al seguente listato:

```
100 on error goto 160
110 input"dividendo";x
120 input"divisore";y
130 z=x/y: rem errore?
140 print"il rapporto è"x/y
150 print: goto 110
160 print "errore"er
170 print "in linea"el
180 print: resume 110
```

Tentando nuovamente di effettuare una divisione per zero, questa, ovviamente, non verrà effettuata, ma il programma non verrà interrotto.

La riga 100, infatti, informa il Gw-Basic che, nel caso in cui si verifichi un errore (qualunque esso sia), deve "saltare" ad eseguire la riga 160. In questa verranno inseriti opportuni comandi che possano aiutare l'utente a capire l'errore commesso: la variabile ER (linea 160) informa sul tipo di errore commesso; nel caso specifico compare 20 che è, appunto, il codice per "Division by zero". Alla variabile EL, invece, è riservato il compito di indicare in quale linea si è verificato l'errore stesso.

L'istruzione Trap, infine, comunica al programma in che punto deve riprendere l'elaborazione dopo il verificarsi dell'errore.

Non è possibile, come è intuitivo, inserire, per ciascun programma, più di una riga contenente il comando "On Error Goto"; tentando di farlo, comunque, verrà reso attivo l'ultimo comando "On Error Goto" incontrato nel corso dell'elaborazione.

E' invece possibile prevedere più istruzioni Trap alle quali accedere, ad esempio, con If...Then dopo aver individuato il tipo di errore occorso.

## Qui C-16

☐ **Da quando ho acquistato la cartridge di espansione per C/16 ho problemi di caricamento con il registratore.**
*(Claudio Bottini - Genova)*
☐ *Ho il C/16 e vorrei alcune informazioni: conviene acquistare l'espansione Ram? Esistono routine grafiche per il mio computer?*
*(Biagio Amato - Palermo)*

• Non posseggo l'espansione per il C/16 e non so fino a che punto sia affidabile (e conveniente). Non esistono routine grafiche per il C/16 perchè il computer possiede di già un set piuttosto sofisticato e completo di istruzioni grafiche.

Per ciò che riguarda il futuro del C/16, bè... caliamo un velo pietoso.

## Gw-Basic che non parla

☐ **Posseggo i vostri prodotto Gw-Basic e la "Voce III", ma non riesco a farli funzionare contemporaneamente. Come mai?**
*(Gabriele Oldano - Ciriè)*
*(Luca Campana)*

• I due programmi occupano le stesse locazioni di memoria e, di conseguenza, è possibile caricarli solo uno alla volta.

## Inalberato nero

☐ **Vorrei ospitalità sulla vostra rivista per le mie lamentele (segue elenco di inconvenienti riscontrati su prodotti Commodore e non).**
*(Maurizio M. - Ambivere)*

• Io comprendo perfettamente il disagio in cui viene a trovarsi chi crede di aver diritto a qualcosa che, invece, gli viene negata.

Tuttavia proverò a spezzare qualche lancia in favore delle Ditte prese di mira dal nostro arrabbiatissimo lettore.

La stampante Star SG-10C, da me stesso recensita nel lontano gennaio '86 (CCC N.27) è ancora quella che, da quella data, utilizzo ininterrottamente per stampare i listati e le schermate grafiche che vedete pubblicati. Ho avuto occasione di provarla con più di un programma grafico ed ha sempre funzionato egregiamente dimostrando, per quello che ne so io, la sua completa e totale compatibilità con il software del C/64.

E' anche vero che mi è capitato di non vederla funzionare con alcune *copie di programmi pirata mal effettuate* ma, in questo caso, non si può certo attribuire la colpa alla stampante che, con il software originale, funziona correttamente.

Analoga correttezza di funzionamento ho riscontrato con l'altra stampante importata dalla Claitron (NL-10) che molti nostri collaboratori posseggono.

E non pensi, il nostro lettore, che questa sia da considerare una "sviolinata" di circostanza nei confronti di un nostro inserzionista: abbiamo dimostrato in più di una occasione di non esser tanto teneri nemmeno con la stessa Commodore di cui ci sentiamo, a ragione, interlocutori "privilegiati". Si tenga presente, in ogni modo, che la Star SG-10C non viene più prodotta.

Per ciò che riguarda la Commodore, posso esser d'accordo sull'avari-

zia di informazioni a proposito del CP/M del C/128; ma da qui a dire che dovrebbe essere obbligata a fornire i (voluminosi) manuali del sistema operativo in questione, ce ne passa! Non dimentichiamo, infatti, che la Commodore ha avuto il solo diritto alla riproduzione dei dischetti del sistema, dal momento che chi detiene il marchio originale è la Digital Research. Anche noi siamo spesso polemici sulla reperibilità di ulteriori informazioni, ma non ci siamo mai sognati di chiederle gratis: una manualistica completa sul C/64 (o, peggio, del C/128) costerebbe sicuramente più del computer stesso.

Sono, però, completamente d'accordo con il lettore sul fatto che i manuali dovrebbero essere tradotti in (comprensibile) italiano. Ma sembra che, finalmente, le cose stiano cambiando. Un po' di pazienza e vedrete che tutto si sistemerà nel migliore dei modi.

Mi permetto di ricordare, dato che siamo in argomento, che una trattazione sufficientemente dettagliata sui comandi del CP/M è stata pubblicata in tre puntate successive sull'altra nostra rivista "Personal Computer" (N. 11, 12, 13). Per chiedere gli arretrati telefonate al solito numero (02/84.67.34.8) chiedendo della signora Lucia.

## News Room e 1526

**□ Posseggo una stampante 1526, modificata 803, tento di usarla con il programma News Room per C/64, ma non riesco a stampare il foglio completo come quello pubblicato a pag.28 di CCC N.34. Forse quello pubblicato è un "collage" di varie parti?**
*(Aldo Satta - Firenze)*

• No, lo giuro sul mio onore: l'immagine citata è stata proprio costruita con l'opzione "Layout" del programma News Room. Io scarterei l'ipotesi della incompatibilità della stampante e opterei per una imperizia d'uso o per una copia mal effettuata del programma stesso.

A onor del vero il programma l'ho provato con una Star SG-10C e con C/64 dotato di Speed Dos. Con un normale 1541 il tempo richiesto per l'edizione di una pagina risulta enormemente lungo. Talmente lungo che conviene creare i vari "pezzi" singolarmente, realizzare un collage e fare una fotocopia: non dimentichiamo che un computer serve solo se fa risparmiare tempo, altrimenti è bene ricorrere a metodi più rozzi (ma veloci) pur se poco... informatici.

## Quantità e prezzo

**□ Un utente di Plus/4 paga i (pochi) programmi che pubblicate per tale computer circa 2000 lire l'uno. Il possessore di C/64, invece, trova tanti articoli per cui ciascuno di questi gli viene a costare circa 400 lire. Vi embra giusto?**
*(Gabriella Mai - Avezzano)*

• Non sono completamente d'accordo con i calcoli effettuati (non si tiene conto dei programmi "universali") ma, comunque, ringrazio la nostra lettrice per l'estrema cortesia con cui ha lanciato l'imbarazzante accusa.

Il ragionamento proposto non fa una grinza, ma vediamo la faccenda da un altro punto di vista: supponiamo di invertire la tendenza e di pubblicare più articoli per C/16 e Plus 4 che non per C/64. Le vendite diminuirebbero a tal punto per cui saremmo costretti a quintuplicare il prezzo di copertina. Il rapporto resterebbe invariato...

## Schemi elettrici

**□ Come posso procurarmi lo schema elettrico del C/64?**

• La guida di riferimento del programmatore ha in appendice ciò che cerchi; è bene tener presente che vi sono molti esemplari di C/64 che si differenziano dallo schema pubblicato, pur se in minimi particolari.

## Poche ma buone

**□ Ho solo poche domande da porre: che cos'è lo Status, lo Stack e il Flag di stato del linguaggio macchina?**
*(Michele Zafarana - Asti)*

• Hai detto niente! Per rispondere esaurientemente sarebbe necessario un intero fascicolo (o quasi) che, del resto, è già stato pubblicato.

Mi riferisco, infatti, al fascicolo "Speciale Commodore" che contiene due trattazioni: la prima è un minicorso di Assembly - Linguaggio Macchina; la seconda è un'applicazione pratica del corso, vale a dire le routine grafiche di Toma corredate di disassemblato commentato.

Puoi chiedere il fascicolo (e, se lo desideri, anche il dischetto) telefonando al servizio arretrati (02/84.67.34.8, signora Lucia).

## Megahertz sarà lei

**□ Che cosa sono i Mhz ed i cili macchina?**
*(Anonimo del tardo '900)*

• Il megahertz è l'unità di misura della frequenza e rappresenta, esattamente, un milione di oscillazioni effettuate in un secondo.

Se, quindi, un microprocessore è in grado di eseguire compiti alla velocità di 7 Mhz, vuol dire che i suoi circuiti sono in grado di eseguire 7 milioni di operazioni elementari al secondo.

Ciò non vuol dire, però, che esegue 7 milioni di istruzioni in L.M nè tantomeno in Basic (che è formato, a sua volta, da gruppi di istruzioni in L.M.): con il termine "operazione elementare", infatti, si intende quella eseguita a livello elettronico, vale a dire quello che è alla base del cosiddetto "ciclo di macchina".

Per esempio l'istruzione LDA#, che nel linguaggio Assembly del 6502 significa "carica l'accumulatore con il numero che segue", richiede due cicli macchina mentre la stessa istruzione, nel formato di indirezione rispetto al registro X, richiede ben sei cicli. Le istruzioni del 6502 richiedono un minimo di due cicli ed un massimo di sette, a seconda dell'istruzione stessa e del suo formato.

Conoscere il numero di cilci richiesti dalle varie istruzioni è indispensabile, per esempio, per sapere il tempo totale di una routine che deve essere inserita nel ciclo di Interrupt. Se il tempo richiesto per la sua esecuzione

# I capolavori Star si adattano ad ogni cornice!

Solo con una stampante di elevata qualità si possono sfruttare ed esaltare tutti i vantaggi di un computer di elevata qualità. Star producè stampanti per computer che rispondono ad ogni tipo di esigenza. Ogni stampante Star è un capolavoro che associa robustezza a precisione, poichè Star è un maestro nella sua arte! Ogni capolavoro Star regge qualsiasi tipo di confronto in quanto a tecnologia, prestazione e prezzo. Belle o veloci, flessibili o specializzate, le stampanti Star si adattano ad ogni computer, poco importa come si chiami e dove si trovi! Dunque, non si affanni a cercare oltre, ma entri anche Lei nella galleria del nostro rivenditore di zona, dove troverà sicuramente la stampante che cerca, unitamente a tanti buoni consigli: siamo certi che anche Lei concluderà che, **con una Star si può andare molto lontano!**

Via Gallarate, 211 20151 Milano
tel. 02/301.00.81 r.a. 301.00.91 r.a.

Per avere maggiori informazioni e l'indirizzo del rivenditore della Sua zona ci invii il coupon allegato.

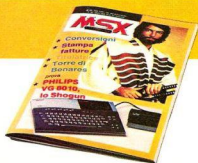
Ditta: ____________ Via: ____________ n° ______

Nome: ____________ Cap.: ______ Città: ____________

____________ Tel.: ____________

# 128 KBYTES

## SINCLAIR COM

## COMMODORE

## MSX

# Il grande software made-in-Italy

### LA VOCE III

Fa parlare e cantare il C64 secondo come lo programmi senza l'uso di campionatori nè sintetizzatori.
Tutte le parole o le canzoni così prodotte possono essere inserite come stringhe in altri programmi.

**Lire 12.000**

### RAFFAELLO

Un programma per disegnare col tuo Comodore 64 col solo joystick senza Koala nè tavoletta grafica.
Tutti i disegni prodotti possono essere memorizzati ed uti lizzati in altri programmi.

**Lire 10.000**

### OROSCOPO

Fa in maniera scientifica l'oroscopo personale. Il più completo programma astrologico per Commodore 64.

**Lire 12.000**

### COMPUTER MUSIC

Un music-editor avanzato più per un programma juke-box con 27 motivi celebri di musica classica e leggera da Arcadia a Bach, Vivaldi, Zeppelin...

**Lire 12.000**

### GESTIONE FAMILIARE

Tre programmi su cassetta che giustificano l'aggettivo "domestico" del tuo computer:
- bilancio familiare;
- dieta equilibrata;
- scheda medica familiare.

Gira su C/64/128

**Lire 12.000**

### BANCA DATI

Un potente data base per C/64 e Spectrum disponibile anche su disco. L'edizione su cassetta contiene da un lato la versione C64 e dall'altro la versione Spectrum.

**Lire 12.000**

### DICHIARAZIONE DEI REDDITI (740/S)

Programma aggiornato al 1986 per la dichiarazione dei redditi, modello semplificato. Per C64.

Disco: **Lire 24.000**
Cassetta: **Lire 16.000**

### MATEMATICA FINANZIARIA

Pubblicato a puntate su Commodore (n.ri 13, 14 e 15) e su Personal Computer (n.ri 1, 2, 3 e 4) questo programma offre un vero e proprio corso completo di ragioneria su Commodore 64.
Se ne consiglia l'acquisto insieme agli arretrati delle riviste che ne illustrano l'uso ed il funzionamento.

Disco: **lire 20.000**
Cassetta: **lire 10.000**
Comodore 13, 14 e 15 e Personal Computer 1, 2, 3
**Lire 21.000**

### ANALISI DI BILANCIO

Naturale completamento di "Matematica Finanziaria" questo programma consente di calcolare automaticamente tutti i ratio più significativi e di confrontare due bilanci dello stesso ente.
Il testo esplicativo è stato pubblicato su Personal Computer n.ri 2, 3, 4 e 5 che si consiglia di acquistare contemporaneamente.

Disco: **Lire 20.000**
Cassetta: **Lire 10.000**
Personal Computer 2, 3, e 5: **Lire 12.000**

### ARREDARE

Un programma professionale per ottimizzare le soluzioni d'arredamento della vostra casa.
N.B. gira solo sotto Simon's Basic.

Disco: **Lire 20.000**
Cassetta: **Lire 10.000**

### GRAPHIC EXPANDER 128

Un potente programma grafico per il c 128 in modo 128.

**Lire 27.000**

---

Sì, inviatemi i seguenti software al prezzo di listino + Lire 3.000 per spese di spedizione:

☐ RAFFAELLO
☐ LA VOCE III
☐ OROSCOPO
☐ COMPUTER MUSIC
☐ GESTIONE FAMILIARE
☐ BANCA DATI
☐ MATEMATICA FINANZIARIA/DISCO
☐ MATEMATICA FINANZIARIA/CASS.
☐ MATEMATICA FINANZIARIA/RIVISTE
☐ ANALISI DI BILANCIO/DISCO
☐ ANALISI DI BILANCIO/CASS.
☐ ANALISI DI BILANCIO/RIVISTE
☐ DICHIARAZIONE DEI REDDITI/DISCO
☐ DICHIARAZIONE DEI REDDITI/CASS.
☐ ARREDARE/DISCO
☐ ARREDARE/CASSETTA
☐ GRAPHIC EXPANDER/DISCO

Valore complessivo: Lire..........................................................

Su tale importo mi praticherete lo sconto del 10% in quanto abbonato a ☐ *Commodore Computer Club* ☐ *Personal Computer* ☐ *Computer* ☐ *VR Videoregistrare*. Pertanto vi invio la somma soltanto di lire..........................................................

☐ Desiderando ricevere le copie ordinate con la massima urgenza, accludo assegno bancario n.ro..........................................................
Banca.......................................................... per lire.......................................................... voi intestato.

☐ Contentandomi dei normali tempi postali ho inviato oggi stesso l'importo di lire.......................... a mezzo C/C postale N. 37952207 intestato a Systems Editoriale - V.le Famagosta, 75 - 20142 Milano.

Ritagliare e spedire in busta chiusa regolarmente affrancata a Systems Editoriale - V.le Famagosta, 75 - 20142 Milano.

**Nome** ..........................................................
**via** .......................................... **N.ro.** .......................................... **telefono** ..........................................
**CAP** .......................................... **Città** ..........................................

aumenta troppo, si corre il rischio di veder girare solo i programmi inseriti nel ciclo di Interrupt e mai quello "principale".

## Che afa fa

□ **Dopo alcune ore di funzionamento del mio computer lo schermo scompare e insorgono vari inconvenienti.**
*(Guglielmo Pardo - Genova)*

• Anche il mio C/64, soprattutto nel periodo estivo, tende ad analoghe sorprese. Il motivo è dovuto alla scarsa ventilazione dovuta all'esiguo spazio presente tra le feritoie inferiori del computer e la superficie di appoggio (di solito un tavolo di legno, per giunta ottimo coibente termico!). Se durante il lavoro il computer è colpito anche dalla luce del sole è inevitabile il verificarsi di vari inconvenienti.

Durante periodi particolarmente caldi, quindi, ti consiglio di allontanare il computer di qualche centimetro dal piano di appoggio per mezzo di opportuni distanziatori (anche semplici bacchette di legno) prestando cura affinchè questi non ostruiscano le già piccole feritoie di areazione!

## Che fare?

□ **Dovrei far riparare il mio C/64 che funziona con programmi che lo fanno parlare ma tace con programmi musicali. Poichè hanno chiesto 100.000 lire per la riparazione del SID, mi conviene farlo riparare oppure è preferibile acquistare il C/128?**
*(Elio Picarone - Napoli)*

• Mi sembra molto strano che il Sid funzioni solo con programmi di sintesi vocale e si rifiuti di farlo con "semplici" programmi musicali. Io ci penserei due volte prima di mandarlo a riparare e proverei routine "sonore" di vario tipo per sincerarmi della presenza di un guasto.

Per ciò che riguarda il C/128, che posso dire? Vi sono dei programmi professionali in giro ma, in fin dei conti, non offrono molto di più di ciò che offre il solo C/64.

Le potenzialità sonore, comunque, sono perfettamente identiche nei due computer. Con il C/128, semmai, potrai scrivere listati molto più lunghi servendoti di comandi Basic molto semplici; gli effetti che si possono ottenere sul C/128 e C/64, invece, sono rigorosamente eguali.

## Complimenti servili

□ **Sono un vostro nuovo lettore e vi pongo alcune domande...**
*(Angelo D'Amico - Cava dei Tirreni)*

• I complimenti di cui è imbottita la simpatica lettera mi hanno fatto quasi arrossire di contentezza: mi piacerebbe farla leggere ai responsabili di quel partito politico che (inverosimile!) si sono rifiutati di accettare la mia candidatura alle passate elezioni.

Scherzi a parte, passiamo alle risposte che, con abile mossa padronale, passo a comunicare:

Le famose routine di Toma sono routine grafiche per il C/64 che consentono di disegnare in prospettiva. Mi sembra, però, che non siano adatte a chi è "molto"

principiante: se vuoi rischiare, puoi richiedere il fascicolo "Commodore Speciale" (che è ad esse dedicato) al nostro servizio arretrati (02/ 84.67.34.8 chiedendo della signora Lucia, alla quale fischieranno le orecchie dal momento che la nomino una risposta sì ed una no).

Il Monitor incorporato nel C/128 è un programma che, seppur rudimentale, consente di lavorare in linguaggio macchina. Tieni presente che nello stesso fascicolo accennato è presente anche un breve corso introduttivo al linguaggio macchina. Che pretendi di più dalla vita? Due argomenti su un solo fascicolo: approfittate siore e siori...

## Cobol e 1541-compatibile

□ **Esiste il Cobol per il C/64? E' valido il disk drive recensito sul N.42?**
*(Mario Messiano - Napoli)*

• C'era una volta la valvola termoionica, ma un bel giorno fu inventato il transistor e, poco dopo, il computer; a qualcuno venne in mente di scrivere un linguaggio in grado di utilizzarlo: nacque il Cobol.

E fu così che, tanti anni fa, il Cobol si diffuse, soprattutto per programmi finanziari e di contabilità.

Da allora decine di linguaggi furono creati per le varie esigenze. Il Cobol sopravvisse perchè in questo linguaggio era stato investito moltissimo denaro, ma anche perchè era stato installato in tante di quelle macchine che si ritenne fosse più economico farlo sopravvivere.

Il Cobol deve essere considerato un animale informatico in via di estinzione, protetto da associazioni bancarie dure di comprendonio.

Non so se esista una versione Cobol per C/64, ma anche se esistesse non ti illudere di poter caricare i programmi che, forse, ritieni di poterti procurare. In ogni caso, penso che nessuno potrà mai aiutarti a trovare una documentazione valida per un suo uso ottimale. Ma consentimi di infierire: il Basic V.2 del C/64 è certamente più valido del brontosauro Cobol, nato in altre epoche e per altri scopi.

E veniamo alla seconda domanda; premetto che le recensioni che vedi pubblicate sulla rivista corrispondono ai risultati di prove piuttosto approfondite e non a non meglio definiti atti di vassallaggio nei confronti dei nostri inserzionisti.

Se affermiamo che un certo prodotto (hardware o software che sia) può esser considerato valido, lo diciamo con ragione, pur se limitatamente alla nostra personale esperienza sul prodotto.

Sembra, del resto, che il drive che tu citi sia venduto non solo da altre Ditte italiane, ma in tutto il mondo: sulla rivista Run (americana) del mese scorso, sono almeno una dozzina le Ditte che vendono per corrispondenza proprio quel drive al prezzo di circa 150 dollari; sulla stessa rivista compare la proposta di un altro modello, sempre 1541 compatibile, di forma diversa e prezzo maggiore.

Con ciò voglio solo ribadire il fatto che con la moderna tecnologia è possibile realizzare imitazioni e compatibilità a prezzi contenuti; il motivo per cui il modello originale abbia prezzi decisamente maggiori può es-

ser dovuto a motivi di politica commerciale:

Ne approfitto per ricordare che alcuni esemplari del modello provato, recentemente diffusi sul nostro mercato, sono privi del deviatore esterno per cambiare il numero del drive; all'interno, comunque, sono sempre presenti i pin che permettono l'applicazione dei deviatori.

Per il resto non posso che confermare la positiva opinione sulla periferica recensita.

## Stampante che s'inceppa

**☐ Spesso la mia stampante, dopo aver stampato qualche foglio, strappa i bordi forati del foglio successivo provocando un disastro.**

• Le moderne stampanti hanno la possibilità di selezionare due tipi di carta: quello cosiddetto "continuo" (rulli, pacchi, eccetera) e quello a fogli singoli.

Il primo tipo è comodissimo quando si devono stampare molte copie di uno stesso documento, listati lunghi, disassemblati e così via. La necessità di inserire il foglio singolo, invece, si verifica quando si debba usare carta prestampata (moduli), carta riciclata per brutta-copia (come il retro di fotocopie che non servono più), carta bollata, fogli protocollo e fogli di formato, comunque, che non si trova facilmente in commercio sotto forma di rotoli continui.

Per consentire all'utente di usare sia l'uno che l'altro tipo di carta, i fabbricanti di stampanti dotano i loro prodotti di una leva che, a seconda di come viene posizionata, rende attiva, o meno, la frizione del rullo. In pratica, con la leva selezionata in quest'ultimo modo, il foglio inserito viene spinto in avanti grazie al contatto forzato tra due rulli di cui, in genere, è visibile solo quello superiore (quello "grosso", per intenderci).

Con il funzionamento a trattore, invece, il rullo non preme contro la carta, che risulta libera, ma il movimento di avanzamento viene affidato ai cingoli chiodati del trattore che, agendo sulle parti laterali forate del foglio, assicurano un corretto funzionamento.

Mentre, però, il rullo viene disinserito, o meno, a seconda della posizione della leva, i cingoli sono sempre attivi. Ciò comporta il fatto che adoperando un pacco continuo di fogli forati lateralmente, l'avanzamento deve essere assicurato SOLTANTO dai cingoli. Spesso, purtroppo, l'utente dimentica di disinserire il rullo e di conseguenza il foglio viene spinto in avanti sia dai cingoli che dal rullo. Poichè la velocità di avanzamento dei due sistemi non è rigorosamente eguale, il foglio viene trattenuto dal rullo mentre i bordi tendono ad andare più velocemente (o viceversa, a seconda della macchina) provocando l'inconveniente lamentato.

Morale: usando fogli continui forati ai lati (che richiedono, quindi, l'uso del trattore) ricordarsi di disinserire il rullo; ti puoi accorgere che questo è realmente disinserito se, inserendo un foglio nella macchina, questo si muove con la massima libertà, senza nessun impedimento; se, invece, sei costretto ad azionare la manopola per spostarlo, vuol dire che è attivo il rullo di frizione.

## Ironie

**☐ Smettetela di pubblicare centinaia di programmi per C/16, Plus/4 e C/128: soprattutto per quest'ultimo riteniamo che vi siano troppi programmi specifici e quasi niente per C/64. E' ora di finirla!**

*(Gigi Leonardi & Gianluca De Crignis - Udine)*

• Avete ragione, e provvedo subito: come potete notare, infatti, da questo numero inizieremo a pubblicare un maggior numero di articoli per il denigrato C/64...

## Modelle e collant

**☐ Suggerirei alle vostre modelle (vedi copertina N.43) di togliersi il collant almeno un'ora prima di fare le fotografie, altrimenti resta il segno dell'elastico.**

*(Marco Zotta - Milano)*

• Provvederò di persona ad esaminare minuziosamente il corpo delle modelle (prima, durante e dopo le foto) affinchè non si verifichino mai più inconvenienti del genere.

## Risposte rapide

## Amiga e Dps-1101

In teoria sarebbe possibile collegare, tramite interfaccia, l'Amiga alla stampante a margherita Commodore Dps-1101 che, però, non potrà mai riprodurre schermate in alta risoluzione. Dubito, tuttavia, che l'operazione sia economicamente conveniente rispetto all'acquisto di una stampante ad aghi specifica.

*(Carlo Calcaterra - Salerno)*

## Mandelbrot

Sono contento che, grazie alla nostra rivista, tu abbia imparato a realizzare programmi grafici interessanti. Telefonaci per concordare un'eventuale collaborazione.

*(Mauro Ottaviani - Roma)*

## Inibizione

Non è possibile inibire il comando Sys 64738 dal momento che la routine, oggetto della Sys, è allocata in Rom e, quindi, sempre disponibile non appena si accende la macchina (guai se non lo fosse!).

*(Mario Tramma - Civitavecchia)*

## Geos su cassetta?

Non è disponibile il Geos su cassetta per il semplice motivo che, durante il suo funzionamento, utilizza il drive decine di volte e sarebbe assurdo effettuare le stesse operazioni con il nastro, a causa dell'estrema lentezza della sua gestione.

*(Massimo Carraro - Piave di sacco)*

## Anti-reset

Abbiamo parlato più di una volta circa la possibilità di annullare l'effetto dei tasti Run/Stop e Restore oppure

H RDWARE SOFTWARE
TELEMATICA HOBBISTICA

20155 Milano - Via Mac. Mahon, 75 - fel. 02/32.34.92 - 32.70.226

**È DISPONIBILE TUTTA LA GAMMA COMMODORE AMIGA 500-1000-2000 CON I PREZZI MIGLIORI D'ITALIA - TELEFONARE!!!**
**RICORDA CHE ALLA NUOVA NEWEL TROVI TUTTO CIÒ CHE DESIDERI PER IL TUO COMPUTER CON PREZZI D'IMPORTAZIONE**

**BANCA DATI MODEM-SHOP - NUOVA NEWEL E IN FUNZIONE TUTTI I GIORNI DALLE 13 ALLE 9 allo 02/32.70.226 300 / 1200 BAUD**

**DISPONIBILI PIÙ DI 1000 PROGRAMMI PER COMMODORE AMIGA!!! RICHIEDETE I CATALOGHI**

DRIVE AGGIUNTIVO PER AMIGA NUOVO MODELLO "SLIM" L. 299.000

DRIVE PER COMMODORE 64/128 TURBO MODELLO "SLIM" L. 299.000

**HARDWARE PER COMMODORE 64/128**

## HARDWARE PER COMMODORE AMIGA

INTERFACCIA MIDI L. **110.000** per collegare una tastiera musicale al tuo amiga
DIGISOUND L. **190.000** per digitalizzare la tua voce e/o suoni favoloso!
DIGIVIEW L. **190.000** per digitalizzare le tue più belle immagini!!!
VIDEOSOUND »»**NOVITÀ 87'**«« L. **299.000** Eccezionale novità un digitalizzatore sonoro e video alla volta un unico apparecchio compatto.
ESPANSIONE 2 MEGA per amiga 1000 (passante) L. **899.000**
ESPANSIONE 1/2 mega per commodore amiga 500/2000 L. **telefonare**
KICKSTART SU ROM 1.2 PLUS + ESP. 256 per amiga 1000 L. **telefonare**
SCHEDA XT-BOARD per AMIGA 2000 ed HARD-DISK 20 MB **disponibili.**

## NOVITÀ HARDWARE PER COMMODORE 64/128

**STARDOS NEW!** Eccezionale novità un velocizzatore che supera persino la velocità dello speed-dos attiva i tasti funzione ecc. In una sola Eprom kit da inserirsi nel c64 con manuale in ital. Non necessita di elaborazioni al drive ne del cavo parallelo. In dotazione anche un disco copiatori velocissimo!!! L. **39.000**
**CARTRIDGE 80 col.** Permette di visualizzare le 80 colonne sul 64. L. **39.000**
**VIDEODIGITAL 64** Nuovo visualizzatore in cartuccia, digitalizza le tue più belle immagini con l'aiuto di una telecamera o videoregistratore semplicissimo da usare con manuale in italiano. Inoltre è possibile modificare le immagini con il KOALA ecc. L. **90.000**

**SUPPORTI MAGNETICI DI PRIMISSIMA QUALITÀ - DISCHETTI DOPPIA FACCIA - DOPPIA DENSITÀ CERTIFICATI (100 pezzi min)**
**Floppy disk "BULK" 5 1/4 ds dd 100% error free da L. 1.200 - Floppy disk "BULK" 3 1/2 ds dd error free da L. 3000**

**SPEEDDOS PLUS (vers. lusso) da L. 49.000**
Il più collaudato ed economico velocizzatore via harware in rapporto prestazioni/prezzo per il sistema (C64/1541), legge 202 blocchi da disco in 20 secondi, tasti funzione, hard copy del video, comandi dos diretti (turbo), completamente trasparente!!!

**DOUBLE SIDE KIT (NOVITÀ)**
L. **8.000**
per scrivere sulla seconda faccia del disco senza più forarlo!!!

**THE NEW FINAL CARTRIGE III per 64/128 (modo 64)** L. **90.000**
L'evoluzione continua!!!
Eccovi l'ultima release della mitica cartuccia notevolmente migliorata e modificata. Turbo, la favolosa routine dello speeddos su cartuccia fino a 10 volte più veloce sia in lettura che in scrittura!!!
8 Tasti funzione programmati, 24 K ram extra per i prog. in Basic.
Un favoloso Sprotettore di programmi tipo O.M.A. incorporato, dischi e cassette IN UN SOLO FILE!!! (+ boot se necessita) Inoltre ha incorporato il GAME KILLER (Evita la collisione dei sprite, ed ben 40 comandi Basic Turbo a disposizione... HARDCOPY "HR" Si premendo un solo tasto potrete fare l'hardcopy del video in 12 gradazioni di grigio!!! **ECCEZIONALE!!!**

**PROGRAMMATORE DI EPROM ORIGINALE MERLIN PP64 a sole L. 249.000** Il miglior programmatore di eprom per commodore 64 con soft in cartuccia!!! (programma tutte le eprom fino alle 27256) il tutto corredato di garanzia originale e manuale italiano.

**PROCESSORE VOCALE (VOICESYNTETIC)**
L. **115.000**
Digitalizzatore vocale tipo "Voice Master" notevolmente migliorato composto da interfaccia hardware + microfono, software interamente in italiano con ampio manuale di istruzioni. Incredibile fai parlare, cantare il tuo commodore 64: puoi programmarlo a fin che riconosca la tua voce e ti risponda.

**FILTRO ANTIDISTURBO (universale)**
L. **25.000**
Questo stupendo apparecchio vi aiuterà a risolvere ed a prevenire moltissimi problemi. Ad esempio sbalzi di corrente sono fatali per un computer. Inoltre vi toglie disturbi che possono influire come distorsione del video, problemi di caricamento programmi ecc...(UTILISSIMO)

**O.M.A. PLUS (BANDID 11) 64/128 & 1280**
L. **75.000**
Eccovi l'ultima rivoluzionaria cartuccia sprotettore di programmi, trasferisce IN UN UNICO FILE ricassettabile il 99,99% del software protetto!!! Da nastro a disco. da disco a disco,da disco a nastro,da nastro a nastro **IN TRE MINUTI ESEGUE TUTTO IL LAVORO!!!**

**512 KRAM + DRIVE 880K INCORPORATO BASIC E MANUALI D'USO.** Il tutto ad un prezzo **INCREDIBILE!!!**

**MODEM PER AMIGA 500/1000/2000 completo a sole L. 249.000**
Modem 300 baud autodial autoanser con telefono 10 memorie tastierino professionale completo di cavo di connessione a amiga e di software di gestione tutto con relativi manuali!!

**PENNA OTTICA GRAFICA (Brio Pen Lusso)**
L. **49.000**
Favolosa penna ottica per commodore 64 e 128 (modo 64) completa di Software di gestione grafico sia su cassetta che su disco il tutto è dotato di istruzioni in italiano.

**MODIFICA MPS 802 NEW GRAPHIC PLUS**
L. **35.000**
Eccezionale rende 100% compatibile la tua MPS-802 con tutti i programmi di grafica come (KOALA, PRINT SHOP, GEOS, ecc..) semplicissima da montare, con chiave istruzioni in italiano!!!

**ADATTATORE TELEMATICO COMMODORE 64/128 L. 199.000**
Eccovi il nuovo modem della commodore 300/1200-75 baud autodial autoanser completo di software su eprom in italiano inoltre compreso l'abbbonamento per un anno al videotel ed alle pagine gialle sip (solo questi due servizi superano abbondantemente il costo del modem)!!!!!!!!!!.

**FLOPPY DISK 1581 3½ da 800 K per 64/128 L. 499.000**

**AMIGA 500 IL COMPUTER DEL FUTURO**
**512 KRAM + DRIVE 880K INCORPORATO BASIC E MANUALI D'USO.** Il tutto ad un prezzo **INCREDIBILE!!! L. 999.000**

**MOUSE COMMODORE (sostituisce il joystick)** compatibile geos
L. 139.000

**EPROM NEW GRAPHIC MPS 801**
L. **25.000**
Si sostituisce al generatore di caratteri della stampante MPS-801 per migliorare la leggibilità della scrittura (car. discendenti).

**DISPONIBILI TUTTI I PEZZI DI RICAMBIO COMMODORE 64**
SCONTI PARTICOLARI PER RIVENDITORI E QUANTITATIVITELEFONATE! PER ULTERIORI INFORMAZIONI RICHIEDETE I CATALOGHI PER IL VOSTRO COMPUTER SPECIFICANDO IL SETTORE, INVIANDO L. 1.000 in francobolli. Ricorda che alla **NEVEL** trovi anche tutto per COMMODORE AMIGA 64-128, MSX, SINCLAIR ZX & QL, ATARI ST e PC compatibili......

del tasto di Reset per C/64, C/16; certamente affronteremo ancora l'interessante argomento. Mi sembra, tuttavia, che vi sia un articolo al riguardo proprio sul fascicolo da cui hai tratto la scheda che ci hai spedito...
*(Francesco Bizzini - Catania)*
*(Christian Cataldi - Svizzera)*

## Riviste straniere

Per procurarti riviste straniere di informatica è necessario recarsi da un edicolante più che fornito. Ti suggerisco di tentare presso le edicole delle stazioni ferroviarie delle grandi città.
*(Gabriele Fasone - Casale M.to).*

## Disabilita Clr/Home

Per evitare di usare inavvertitamente il tasto Clr/Home è necessario il ricorso a subroutine di Input controllato, pubblicate in varie versioni sulla nostra rivista.
*(Gabriele Zavenio - Siracusa)*

## Giardini informatici

Dubito che pubblicheremo software relativo al giardinaggio a causa della nostra totale incompetenza in materia. Possiamo, però, esaminare benevolmente eventuali programmi sviluppati dai lettori.
*(Giovanni Lenci - Cappella M.re)*

## Totocalcio su 128

Non è possibile trasportare programmi commercializzati (come il Tot Professional) su computer diversi da quelli per cui sono specificamente previsti. Il comando di formattazione è il seguente:
Open 15,8,15,"n0:nome,id"
*(Patrizio Apicella - Barra)*

## Da C/128 in CP/M

Per utilizzare il C/128 in modo CP/M è sufficiente inserire nel drive il dischetto di sistema e accendere il computer. Non è possibile usare il CP/M con il registratore.
*(Graziano Vignolini - Montemurolo)*

## Penna ottica per C/64

Abbiamo già affrontato l'argomento sulla gestione della penna ottica sul N.30, che puoi richiedere al nostro ufficio arretrati.
*(Gianni Vatenova - Pescara)*

## Informazioni sul 1571

Non esiste un libro che parli diffusamente (e specificamente) del drive 1571.
*(Alfredo Savoldelli - Parre)*

## Gw-Basic

Il nostro emulatore Gw-Basic non consente la lettura di un dischetto formattato con il computer IBM, ma rende possibile digitare su C/64 molti dei programmi che possono girare con il PC.
*(Francesco Venturini - Brescia)*

## E' compilato

Se un programma, alla richiesta del List, sembra formato da una sola istruzione del tipo "Sys XXXX", vuol dire che il programma stesso è compilato, oppure scritto in linguaggio macchina.
*(Aldo Sampietro - Vigevano)*

## Chissà dov'è

Non sappiamo da chi è possibile acquistare giochi su disco per Plus/4. Hai provato a contattare i nostri inserzionisti che vendono software per corrispondenza?
*(Illeggibile - La Maddalena)*

## Koala C/16

Il programma pubblicato non contiene errori di sorta ed il messaggio "Out of memory error", pertanto, è dovuto ad errori di digitazione da parte tua (spazi non richiesti tra varie istruzioni, subroutine che chiamano se stesse e così via).
*(Maurizio Tangarra - Taranto)*

## MSX e C/64

I due computer presentano differenze sostanziali ed è certamente questo il motivo per cui lo stesso programma, digitato su un MSX e su un C/64, non gira allo stesso modo.
*(Antonio Lapresa - Monteparano)*

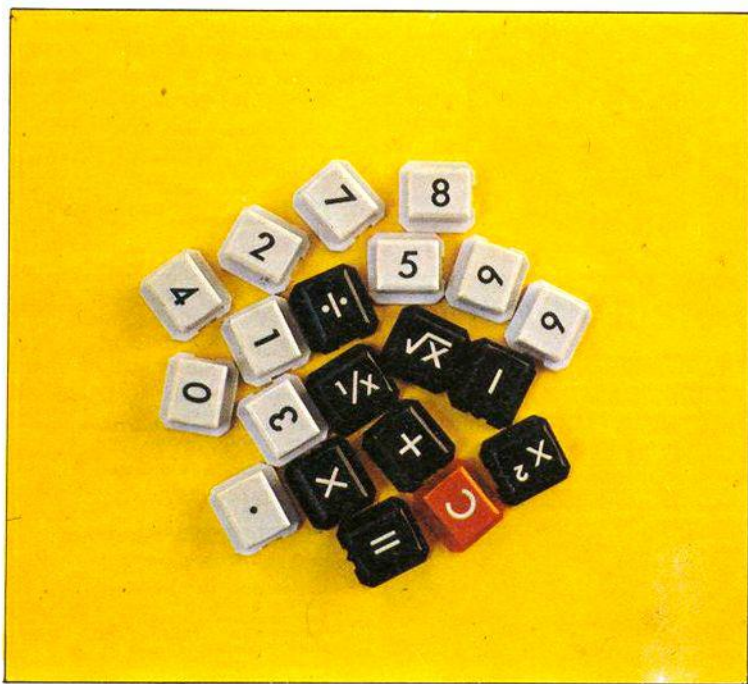
## CHI NON RISICA... NON RISICOM

Per motivi di carattere tecnico, durante la fase di masterizzazione della cassetta **COMMODORE 64 CLUB N. 2**, il 43% circa di tale produzione presenta malfunzionamenti durante l'esecuzione del programma **RISICOM 64**.

Il programma in questione verrà in ogni caso ripubblicato sul N. 3 di **COMMODORE 64 CLUB**

# Tutte le variabili del C/128

*Un'utility indispensabile per facilitare il debug dei listati Basic*

di Giuseppe Mammano

Il Commodore 128 ha un Basic ricchissimo di comandi e funzioni e non necessita, pertanto, di alcuna "espansione" Basic, a parte la mancanza di qualche utility che possa agevolare chi scrive programmi.

Particolarmente utile è il comando DUMP, che permette di visualizzare il contenuto, ed i nomi, di tutte le variabili presenti nel programma stesso.

Capita, infatti, di dimenticare di avere già adoperato una variabile oppure di aver trascritto in maniera inesatta il nome di una variabile.

Sarebbe quindi utile disporre del comando DUMP del quale, in modo 64, esistono decine di versioni.

Le difficoltà incontrate nella stesura del breve listato di queste pagine sono state notevoli, poichè bisogna barcamenarsi tra i banchi in cui è suddivisa la memoria disponibile per il C/128.

La routine effettua il Dump di tutte le variabili, eccetto gli arrays, e si può mandare in esecuzione facilmente inserendo uno o più Stop in punti strategici del listato Basic da controllare e premendo il tasto F1 non appena il programma si blocca. Digitando "Cont" si potrà poi continuare l'esecuzione del programma.

La routine occupa l'area di memoria da $1300 a $13bd (attenzione alle sovrapposizioni con altri programmi eventualmente già presenti nella stessa area).

E' opportuno ricordare (soprattutto ai non esperti) alcune sintetiche note allo scopo di comprendere come il C/128 depositi le variabili e come faccia il programma a visualizzarle.

## Il banco delle variabili

Il Basic fa largo uso delle variabili ma, a differenza del Pascal, non richiede la preventiva dichiarazione nè del tipo nè del nome, operazioni, queste, molto utili per un facile debug e, soprattutto, per la stesura di un listato il più strutturato e lineare possibile.

Le variabili di cui comunemente ci serviamo sono di tre tipi: intere, reali e stringa. Si possono adoperare anchee vettori e arrays, cioè matrici a una o più dimensioni, da dichiarare all'inizio del programma, atte a contenere una serie di dati numerici o letterali.

Non vogliamo, però, occuparci dei vettori, anche perchè il programma, per motivi di praticità, non li visualizza: immaginate, infatti, il dump di una matrice di dimensioni (255, 255): si tratterebbe di visualizzare (se la memoria lo consentisse) ben 65536 variabili per un solo vettore.

Ogni volta che si dichiara una variabile, usando una linea come questa...

*P=3.142122: A$="C.C.C": D%=32*

...il nostro computer deve memorizzarne nome e valore.

La zona di memoria adoperata per "ricordare" le variabili è chiamata VLT (tavola elenco variabil). Nel C/64 questa è una zona di memoria situata subito dopo la fine del programma Basic e la sua posizione varia continuamente se si aggiungono o sottraggono linee Basic. Nel C/128, al contrario, la sua posizione è fissa.

Bisogna infatti ricordare che nel computer in esame esiste un banco di memoria, e precisamente il banco uno, dedicato alla memorizzazione delle variabili. Sappiamo che i 128 kbyte di memoria dichiarati nel prompt di apertura sono suddivisi in due banchi da 64 K (circa).

Di questi, 58109 sono disponibili per i programmi Basic, mentre i restanti sono adoperati come deposito per le variabili.

In particolare, il listato è presente nel banco zero, le variabili nel banco uno, l'interprete Basic si trova invece nel banco quindici. Quest'ultimo, molto acrobaticamente, per mezzo di apposite routine del sistema operativo, salta in continuazione di banco in banco senza che a noi sia dato di rendercene conto.

In realtà si possono notare lievi rallentamenti nell'esecuzione dei programmi.

Provate a digitare questo programma in modo 64 e, subito dopo, in modo 128. La variabile TI$ vi dirà che il C/64 è stato più veloce nell'eseguirlo:

```
10 ti$="000000"
20 For T=1 to 20000: next
30 print ti$
```

Altra notevole differenza tra i due computer è data dalla totale perdita delle variabili non appena, in modo 64, si modifichi una linea di programma e dalla conseguente risposta "can't continue error" alla eventuale richiesta di continuare il programma dal punto di interruzione.

In modo 128 tutto questo non accade: è possibile modificare linee di programma ed impartire un Cont che verrà tranquillamente eseguito.

Ma torniamo alle variabili. Nel C/128 le troviamo immagazzinate nel banco uno a partire da $0400, cosa peraltro testimoniata dalle locazioni $2f, $30, le quali costituiscono, appunto, il puntatore all'inizio delle variabili.

E' possibile sapere dove termina la VLT osservando le locazioni $31, $32 che costituiscono il puntatore all'inizio delle matrici: queste ultime sono infatti immagazzinate subito dopo le variabili "comuni".

Per ogni tipo di variabile (stringa, reale, intera) vengono adoperati 7 byte.

Di questi, i primi due contengono i caratteri costituenti il nome della variabile, i restanti cinque depositano le informazioni sulla variabile.

Come farà la nostra CPU a distinguere tra i tre tipi?

E' presto detto : ricorre ad un trucco di codifica.

Digitiamo le seguenti linee (N.B: [R]= premere il tasto Return):

```
clr [R]
a=0.5 [R]
b$= "prova" [R]
c%=1 [R]
```

(N.B. il comando Clr serve a ripulire la memoria da eventuali altre variabili presenti.)

Entriamo adesso in Monitor (Monitor [R]) e digitiamo:

```
M 10400 10414 [R]
```

Verranno visualizzate le seguenti linee:

```
>10400 41 00 80 00 00 00 00 42
>10408 80 05 f9 fe 00 00 c3 80
>10410 00 01 00 00 00 -- -- --
```

I primi 7 byte si riferiscono ad una variabile reale. Precisamente ad A che è stata dichiarata uguale a 0.5.

Di questi, 41 è il codice ascii della lettera A, 00 è il secondo carattere della variabile (nel caso specifico, infatti, non esiste un secondo carattere).

80 00 00 00 00 sono invece i "codici" che rappresentano il valore della variabile reale, ed esprimono la cifra 0.5 in una notazione particolare, detta notazione esponenziale.

Purtroppo, per comprendere questo tipo di notazione, sarebbe necessario essere versati in matematica e questo non capita tutti i giorni. Le notazioni esponenziali cui siamo abituati sono di solito in base dieci. Ad esempio, nell'espressione 1*10 ↑ 3, equivalente alla cifra mille, il numero 1 rappresenta la mantissa e la cifra 3 l'esponente.

Nel caso del C/128 la base non è dieci, ma due e, dei cinque byte sopra citati, il primo rappresenta l'esponenete aumentato di 128 unità, i restanti quattro rappresentano la mantissa.

Esistono, ovviamente, particolari routine dell' interprete Basic che si incaricano di "tradurre" la notazione esponenziale nei numeri che abitualmente vediamo quando, ad esempio, digitiamo PRINT A.

Alcune di queste routine sono state adoperate nel programma di DUMP.

I successivi sette byte riguardano invece la variabile B$.

Questi sono: 42 80 05 f9 fe 00 00

Quarantadue è il codice ascii della lettera B, 80 rappresenta il secondo carattere della variabile (nel caso specifico lo zero, in quanto la variabile possiede un solo carattere).

Ma perchè $80 (cioè 128 decimale) se il valore dovrebbe essere zero, come per la variabile precedente ?

Il motivo è da ricercare nelle convenzioni cui accennavamo prima.

L'interprete è in grado di sapere che sta lavorando con una variabile stringa proprio per questo motivo. I progettisti della Commodore hanno infatti adottato le seguenti convenzioni:

*Caso della variabile reale: scrive i caratteri del nome con le loro cifre ascii.*
*Caso della variabile stringa: aggiunge 128 al primo carattere del nome e scrive il secondo normalmente.*
*Caso della variabile intera: aggiunge 128 ad entrambe le cifre del nome.*

L'interprete Basic, ed anche il programma presentato in queste pagine, saprà quindi distinguere i tre tipi di variabili.

Se il secondo carattere è maggiore di 128 ed il primo no, si tratta sicuramente di una stringa.

Se ambedue le cifre indicanti il nome della variabile sono maggiori di 128, si tratta di una variabile intera.

Se ambedue i byte sono minori di 128, infine, si è sicuramente in presenza di una variabile reale.

Se poi notiamo che solo i numeri binari maggiori di 128 presentano il bit 7 (cioè quello più a sinistra) posto ad uno, ci renderemo conto che basta esaminare quest'ultimo per conoscere quanto detto prima. Sarà allora sufficiente porre a zero il bit numero 7 per riavere il byte originario.

Ma vediamo un esempio:

*decimale: 128*
*binario: 10000000*
*esadecimale: $80*

Supponiamo di esaminare un byte con valore 128. In esso il bit 7 è posto ad uno, quindi per riavere il byte originario dobbiamo settarlo a zero. Questo si può fare in vari modi.

Nel listato assembly presentato in questo articolo ciò viene effettuato con le seguenti 3 operazioni:

| istr | carry | accum | | carry | accum |
|---|---|---|---|---|---|
| asl | 0 | 10000000 | diventa | 1 | 00000000 |
| rol | $c1 1 | 00000000 | diventa | 0 | 00000001 |
| lsr | 0 | 00000000 | diventa | 0 | 00000000 |

La prima istruzione shifta a sinistra il byte presente nell'accumulatore, in modo che il bit 7 cada nel carry.

La seconda, ruota a sinistra il byte $c1, in modo che il carry vi entri da destra.

La terza istruzione shifta a destra di un bit l'accumulatore fornendo il numero originariamente immesso cui è stato sottratto 128.

Il numero 128 ($80) diventa quindi 00, mentre il byte $c1 indica che cosa è avvenuto.

La CPU, in pratica, incontrando i due byte identificanti il nome della seconda variabile impostata ($42, $80), concluderà immediatamente che si tratta di una stringa (secondo byte maggiore di 127) e che il suo nome è B ($42) seguito da nessun altro carattere: dunque B$.

Per quel che concerne la rappresentazione in memoria del suo contenuto, diremo che il terzo byte rappresenta la lunghezza della stringa stessa, nel caso specifico 5 caratteri.

Dei restanti 4 byte, due sono l'indirizzo al quale cercare il contenuto della stringa e i restanti due sono inutilizzati (esistono perchè è molto comodo codificare tutte le variabili con lo stesso numero di byte).

Se, infatti, (da monitor) digitiamo...

*>1FEF9 [R]*

...avremo il piacere di trovare, nella parte ascii del dump di memoria, la scritta PROVA, cioè la nostra variabile B$.

Per quel che concerne i numeri interi, a questo punto, il discorso è molto lineare: C3 80 00 01 00 00 00, rappresenta la nostra ultima variabile, cioè C%

Infatti $C3, tolto 128 ($80), diventa $43, cioè il codice ascii della lettera C. Sottoponendo $80 alla stessa operazione otteniamo 00, infatti la variabile C% ha il nome formato da una sola lettera.

Dei restanti 5 byte sono adoperati solo i primi due.

I numeri interi sono rappresentati diversamente dai reali. Essi sono infatti memorizzati allo stesso modo degli indirizzi della memoria, con la differenza che, per essi, si deve moltiplicare per 256 il byte più alto.

In questo caso avremo 0*256+1=1.

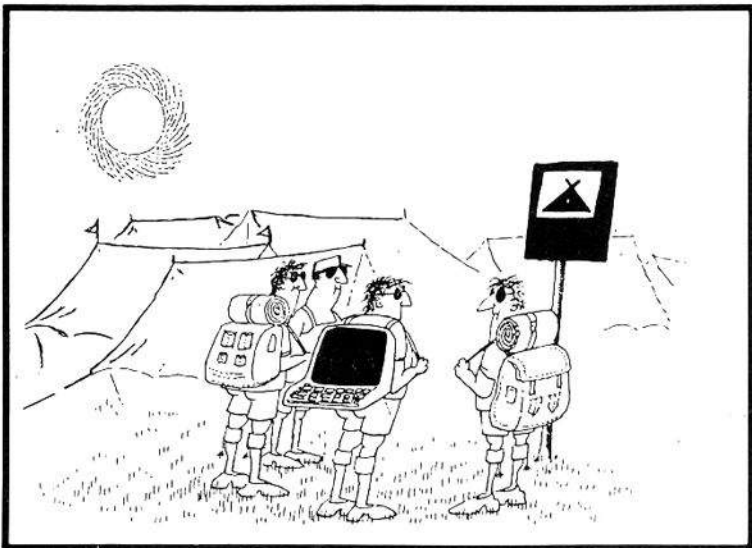
Uno era infatti il valore di C%

Se avessimo invece scritto C%=300, avremmo trovato questi valori:

*C3 80 01 44 00 00 00*

Infatti 256*1+44 è uguale a 300.

Utilizzando solo due byte per la memorizzazione, non si potranno immagazzinare numeri superiori a quello che contiene 255 in ambedue i byte, pari a 255*256+255, cioè a 65536. E' però necessario riservare una metà di questo ristretto intervallo ai numeri negativi. In conclusione, il nostro computer può rappresentare valori interi da -32768 a +32767.

L'uso della routine pubblicata è semplicissimo: tutte le volte che si premerà il tasto F1 (oppure che si impartirà il comando: Sys 4867), verranno visualizzate le variabili, ed il loro contenuto, nello stesso ordine in cui sono state dichiarate nel corso dell'elaborazione.

```
Dump Variabili per C/128 (in modo 128)
         Disassemblato commentato
---------------------------------------
1300 jmp $e386 ;<Ready> fine programma
1303 lda $2f   ;trasferisce i puntatori
1305 ldy $30   ;all'inizio delle varia-
1307 sta $61   ;bili in area di lavoro.
1309 sty $62   ;----------------------
130b cpy $32;  ;compara con puntatori
130d bne $1313 ;inizio schiere.
130f cmp $31   ;se coincidono
1311 bcs $1300 ;il programma e'finito.
1313 ldy #$00  ;-------inizio----------
1315 sty $c1   ;azzera indicatore
1317 iny       ;preleva il secondo
1318 jsr $13b6 ;carattere della varia-
131b asl a     ;bile in esame
131c rol $c1   ;se>127 il bit 7 va nel
131e lsr a     ;carry.Da questo in $c1
131f sta $005c,y;carattere in $5b,$5c
1321 dey       ;preleva il primo carat-
1322 bpl 1318  ;tere e ripete il ciclo.
1325 ldx $c1   ;legge indicatore.
1327 beq $1346 ;se contiene zero e' una
1329 dex       ;v.reale.
132a beq $1384 ;se contiene uno esce.
132c dex       ;se contiene tre e' una
132d beq $135e ;v.stringa.


        -routine variabile intera-

132f ldy #$b0  ;in y il puntatore alla
1331 jsr $1394 ;tabella (%=) e stampa
1334 ldy #$02  ;preleva i due byte del
1336 jsr $13b6 ;valore della variabile
1339 pha       ;e carica con questi
133a iny       ;a ed y.chiama una rou-
133b jsr $13b6 ;tine dell'int.basic che
133e tay       ;converte un numero a 16
133f pla       ;bit in un num.in virgo-
1340 jsr $84c9 ;la mobile.chiama poi la
1343 jmp $1358 ;rout.che stampa il num.


          -routine variabile reale-

1346 ldy #$b1  ;y puntatore tabella
1348 jsr $1394 ;visualizza nome variab.
134b jsr $7c57 ;carica in $49,$4a l'in-
134e lda $49   ;dirizzo del suo valore.
1350 ldy $4a   ;.'
1352 jsr $8ab4 ;carica in acc1.converte
1355 jsr $8c28 ;da virg.mobile a forma-
1358 jsr $8e3c ;to ordinario.visualizza
135b jmp $1381 ;salta al seguito

        -routine variabile stringa-

135e ldy #$b3  ;puntatore tabella
1360 jsr $1394 ;stampa nome e $=
1363 lda #$22  ;codice ascii virgolette
1365 jsr $ef79 ;jsr chrout
1368 ldy #$04  ;preleva lunghezza
136a jsr $13b6 ;e indirizzo
136d sta $25   ;della variabile
136f dey       ;in esame, li deposita
1370 jsr $13b6 ;in $24,$25
1373 sta $24   ;e salta alla subroutine
1375 dey       ;del s.o. che visualizza
1376 jsr $13b6 ;la stringa
1379 jsr $55e8 ;
137c lda #$22  ;chiude le virgolette;
137e jsr $560c ;
1381 jsr $5598 ;va a capo
1384 jsr $13a7 ;salta subrt run-stop
1387 clc       ;azzera il carry
1388 lda $61   ;carica l'indirizzo
138a ldy $62   ;della attuale variabile
138c adc #$07  ;e aggiunge 7
138e bcc $1391 ;se e'il caso incrementa
1390 iny       ;la locazione $61
1391 jmp $1307 ;ricomincia il ciclo

1394 lda $5c   ;--subrt video--
1396 jsr $560c ;stampa il primo
1399 lda $5d   ;ed il secondo carattere
139b beq $13a0 ;(se esiste)

139d jsr $560c ;
13a0 tya       ;ed anche i valori presi
13a1 ldy #$13  ;dalla tabella
13a3 jmp $55e2 ;
13a6 rts       ;
13a7 jsr $f66e ;--subrt run/stop--

13aa beq $13ad ;controlla se e'stato
13ac rts       ;premuto run/stop
13ad jmp $4003 ;in tal caso <READY>
13b0 .byte $25,$3d,$00;--tabella------
13b3 .byte $24,$3d,$00;    (%= $= )
13b6 lda #$61  ;--subrt indfet--
13b8 ldx #$01  ;a=puntatore in pag.zero
13ba jsr $f7d0 ;x=numero banco.preleva
13bd rts       ;un carattere e ritorna
```

---

N.B: la routine "indfet" permette di prelevare un carattere da qualsiasi banco restituendolo nell'accumulatore. Per fare questo bisogna scegliere due byte consecutivi in pagina zero nei quali inserire nel solito modo (byte basso, byte alto) l'indirizzo dal quale prelevarei dati. Nel caso specifico il puntatore e' $61, $62 e contiene l'indirizzo $0400. Prima di chiamare la routine bisogna caricare il registro x con il numero del banco, il registro y con l'eventuale offset e il registro a con il numero del primo dei due byte.

---

```
100 REM DUMP PER C-128 (MODO 128)
110 REM      BY GIUSEPPE MAMMANO
120 REM          CALTANISSETTA
130 :
140 SCNCLR:CHAR 1,12,12,"ATTENDERE...",1
150 COLOR 0,1:COLOR4,1:COLOR1,2:COLOR5,2
160 FOR T=4864 TO 5054
170 READ A:POKET,A:SUM=SUM+A:NEXT
180 IF SUM=19049 THEN 200
190 SCNCLR:CHAR 1,12,12,"ERRORE!":END
200 KEY 1,"RSYS4867■"+CHR$(13)
210 DO WHILE M<50:M=M+1:SCNCLR
220 CHAR 1,12,12,"<F1> =DUMP",1:LOOP
230 SCNCLR:SYS DEC("4003")
240 DATA 76,55,77,165,47,164,48,133
250 DATA 97,132,98,196,50,208,4,197
260 DATA 49,176,237,160,0,132,193,200
270 DATA 32,182,19,10,38,193,74,153
280 DATA 92,0,136,16,243,166,193,240
290 DATA 29,202,240,88,202,240,47,160
300 DATA 176,32,148,19,160,2,32,182
310 DATA 19,72,200,32,182,19,168,104
320 DATA 32,201,132,76,88,19,160,177
330 DATA 32,148,19,32,87,124,165,73
340 DATA 164,74,32,180,138,32,40,140
350 DATA 32,60,142,76,129,19,160,179
360 DATA 32,148,19,169,34,32,121,239
370 DATA 160,4,32,182,19,133,37,136
380 DATA 32,182,19,133,36,136,32,182
390 DATA 19,32,232,85,169,34,32,12
400 DATA 86,32,152,85,32,167,19,24
410 DATA 165,97,164,98,105,7,144,1
420 DATA 200,76,7,19,165,92,32,12
430 DATA 86,165,93,240,3,32,12,86
440 DATA 152,160,19,76,226,85,96,32
450 DATA 110,246,240,1,96,76,3,64
460 DATA 37,61,0,36,61,0,169,97
470 DATA 162,1,32,208,247,96,0
480 END

READY.
```

# A grandi linee

*Un brevissimo programma per studiare la struttura "molecolare" dei caratteri; e alcune utili informazioni sulla loro organizzazione nella ROM*

**di Maurizio Dell'Abate**

**SCHEDA TECNICA**

Software didattico per applicazioni semi-grafiche.

Idoneo per computer C/64, ma adattabile ad altri computer Commodore, pur se con qualche difficoltà.

E' consigliato l'uso della stampante.

Ideale per chi intenda approfondire le proprie conoscenze sulla struttura del computer.

Anche il programma pubblicato in queste pagine è contenuto nel disco "Directory" di questo mese.

Nel Commodore 64 ciascuna cella della pagina testo (byte 1024-2023) può contenere 256 valori (numerati da 0 a 255): si deduce immediatamente che il computer è in grado di visualizzare, contemporaneamente, fino a 256 caratteri diversi.

Il singolo carattere, ma questo dovrebbe esser già noto ai nostri lettori, è composto in una griglia (matrice) di 8x8 punti. Sono dunque necessari 8 byte (valori numerici compresi tra 0 e 255) per configurare ogni carattere.

La mappa caratteri in ROM è la parte della memoria che contiene i dati relativi ai caratteri e viene chiamata, nel gergo, "ROM generatrice"; ha un'estensione pari a 2048 byte, cioè 2 Kbyte.

E' necessario ancora precisare che il C/64 possiede, in ROM, due gruppi da 256 caratteri ciascuno, selezionabili in qualsiasi momento mediante la pressione contemporanea del tasto Shift e del logo Commodore; possiamo quindi concludere che la mappa caratteri in ROM si estende per 4 Kbyte.

Il primo gruppo di caratteri è detto "maiuscolo/grafici": infatti i caratteri che la costituiscono sono soprattutto le lettere alfabetiche maiuscole (naturalmente anche i caratteri numerici) ed i caratteri semigrafici. La seconda serie viene usualmente definita "maiuscolo/minuscolo" perchè è possibile visualizzare contemporaneamente lettere maiuscole e minuscole, oltre ad alcuni caratteri semigrafici.

## Il codice schermo

Ogni carattere visualizzabile è associato ad un codice schermo, ben diverso dal codice Ascii, con il quale i principianti tendono spesso a confonderlo.

Il codice schermo (detto anche "poke code") è un numero compreso tra 0 e 255, indica al computer quale dei 256 caratteri visualizzare e si trova in ciascuno dei 1000 byte riservati alla pagina testo. Infatti se digitiamo...

*POKE 1024,1*

...il carattere corrispondente alla lettera "A" verrà visualizzato nell'angolo in alto a sinistra del video.

Volendo conoscere il Poke Code corrispondente ad un particolare codice Ascii sarà sufficiente digitare:

*PRINT CHR$(147): CHR$(X): PEEK(1024)*

in cui X non è altro che il codice Ascii del carattere desiderato.

Non tutti sanno, però, che il Poke Code di un carattere è la posizione

| SERIE | INDIRIZZO | CARATTERI |
|---|---|---|
| M/G | 53248 - 53759 | Maiuscoli |
| | 53760 - 54271 | Grafici |
| | 54272 - 54783 | Maiuscoli reverse |
| | 54784 - 55295 | Grafici reverse |
| M/M | 55296 - 55807 | Minuscoli |
| | 55808 - 56319 | Maiuscoli e grafici |
| | 56320 - 56831 | Minuscoli reverse |
| | 56832 - 57343 | Maiuscoli e grafici reverse |

Tabella 1

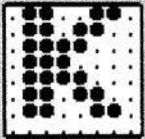
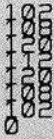
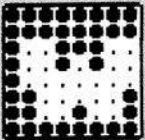
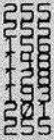
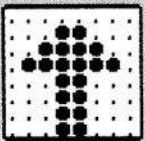
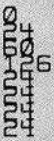
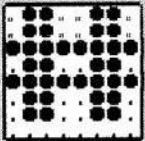
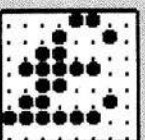
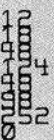
ordinale, all'interno della mappa, degli 8 byte che configurano il carattere stesso.

## La mappa caratteri

La mappa caratteri ha un'estensione di 4 Kbyte ed è situata fisicamente a partire dall'indirizzo 53248 fino a 57343 compreso (per un totale di 4096 byte). I 2 Kbyte della serie "maiuscolo / grafici" si trovano prima dei 2 K riservati alla serie "maiuscolo / minuscolo". Nella tabella è riportata l'organizzazione dettagliata dei 4 Kbyte.

In base a quanto affermato finora, digitando...

*PRINT 53248 + Poke Code*8*

...dovremmo ottenere in risposta l'indirizzo della mappa caratteri dove si trova il primo degli 8 byte che configurano il carattere contrassegnato da Poke Code. Se il carattere appartiene alla seconda serie il risultato va aumentato di 2048 unità.

## Il primo problema

Il C/64 riconosce la memoria 53248-57343 come destinata ad altri scopi: scheda del video e del suono, I/O e altre funzioni. Se infatti andiamo a PEEKare questa zona NON otterremo i dati dei caratteri ROM.

Questa apparente contraddizione si spiega con il fatto che la ROM generatrice di caratteri si trova negli indirizzi suddetti, ma in uno "strato inferiore" (immaginate... in cantina). Sarà sufficiente ordinare al microprocessore di considerare lo "strato" che ci interessa nel modo che vedremo fra poco.

Tuttavia, a causa della scansione della tastiera, il computer accede ogni 1/60 di secondo agli stessi indirizzi, nei quali si aspetta di trovare i registri del chip di I/O.

Se al loro posto trova la mappa caratteri, invece delle istruzioni per la gestione I/O, le conseguenze sono imprevedibili e non certo rosee (leggi: si inchioda brutalmente il tutto). Sarà pertanto necessario disabilitare

VIDEO (V) O CARTA (C)?

VIDEO

CODICE SCHERMO: 18

Figura 1

l'interrupt quando la mappa è selezionata.

Se non avete capito queste ultime righe, non scoraggiatevi, perchè si tratta solo di "teoria"; la pratica viene adesso.

## Lettura della mappa caratteri

Volendo leggere i dati presenti nella mappa caratteri ROM occorre quindi, prima di ogni altra cosa, impartire le due poke seguenti (il tutto deve essere fatto SEMPRE nell'ambito di un programma che gestisca i caratteri):

*POKE 56334, PEEK(56334) AND 254*

...che disabilita l'interrupt, e...

*POKE 1, PEEK(1) AND 251*

...che seleziona la ROM generatrice di caratteri.

A questo punto è possibile leggere (mediante PEEK, se in Basic, oppure LDA, se in Linguaggio Macchina) dalla mappa caratteri (53248-57343) i byte che necessitano.

Gli 8 valori che caratterizzano, per esempio, la lettera "R" maiuscola (Poke Code= 18), vengono resi noti dal ciclo FOR...NEXT:

*S=53248 + 18*8: FOR A=S TO S+7: PRINT PEEK(A): NEXT*

Quando si è terminata la lettura dalla ROM, per riportare il sistema alle condizioni iniziali occorre digitare:

*POKE 1, PEEK(1) OR 4*

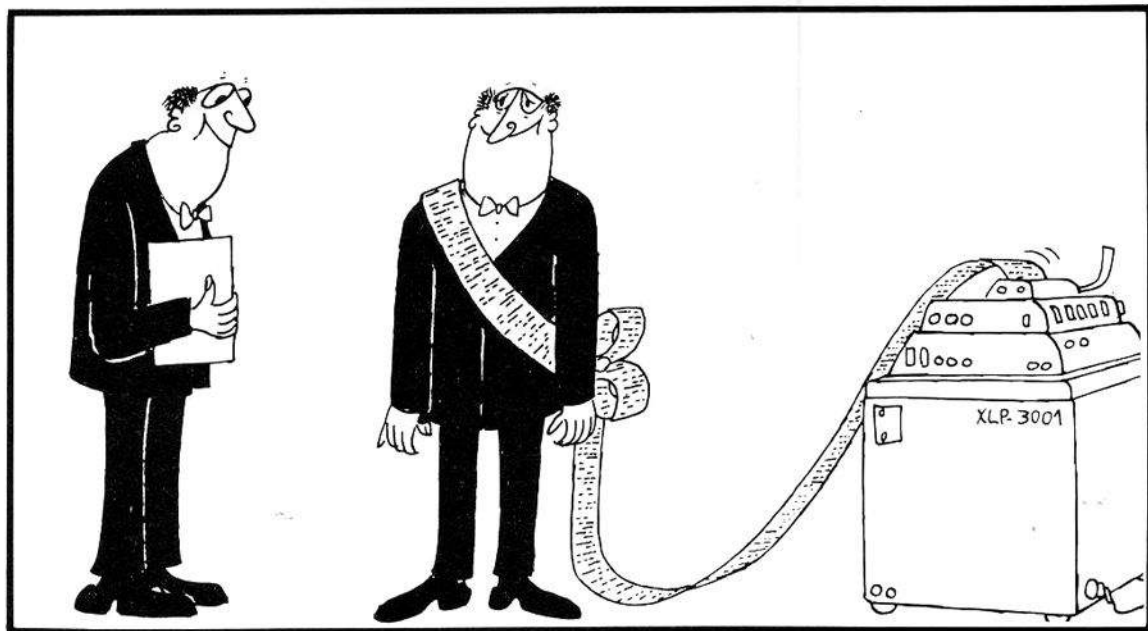
...che seleziona I/O in 53248-57343, e...

*POKE 56334, PEEK(56334) OR 1*

...che ripristina le interruzioni originali.

Un'ultima nota: tutte le operazioni di lettura della ROM generatrice di caratteri devono essere svolte, lo ribadiamo, all'interno di un programma (sia esso in Basic oppure LM) e non in modo diretto: disabilitare l'interrupt comporta, infatti, l'impossibilità di usare la tastiera; in modo diretto, senza tastiera, si fa ben poco!

## Il programma

Il programma pubblicato in queste pagine è un semplice esempio di utilizzo dei dati contenuti nella ROM generatrice di caratteri. Dopo aver scelto uno qualsiasi dei 512 (=256+256) caratteri codificati in ROM, questo viene visualizzato fortemente ingrandito sul video oppure stampato su carta. Il carattere viene visualizzato affiancato dal valore degli 8 byte che lo caratterizzano.

```
100 REM CARATTERI INGRANDITI
110 REM PER COMMODORE 64
120 :
130 ER$="●":CS=81:POKE 198,0
140 PRINTCHR$(147);
150 PRINT"CARATTERI INGRANDITI"
160 PRINT:PRINT"DI MAURIZIO DEL
    L'ABATE"
170 PRINT:PRINT"MAIUSCOLO (G) O
     MINUSCOLO (P)?";
180 GET A$:IF A$="G" THEN PRINT
    " MAI":S=142:GOTO 210
190 IF A$<>"P" THEN 180
200 PRINT" MIN":S=14
210 PRINT"POSITIVO (P) O NEGATI
    VO (N)?";
220 GET A$:IF A$="P" THEN PRINT
    " POS":Z=146:GOTO 250
230 IF A$<>"N" THEN 220
240 PRINT" NEG":Z=18
250 PRINT"PREMI IL TASTO CORRIS
    PONDENTE AL"
260 PRINT"CARATTERE DA OSSERVAR
    E...."
270 GET K$:IF LEN(K$)=0 THEN 27
    0
280 PRINT"[CLEAR][DOWN][RIGHT]"
    ;CHR$(S);CHR$(Z);K$
290 FOR I=1 TO 1000:NEXT:H=PEEK
    (1065)
300 PRINTCHR$(147);CHR$(142)
310 PRINT"[6 DOWN][2 RIGHT]PER
    FAVORE ATTENDI"
320 PRINT"[2 RIGHT]────────────
    ────────":REM TASTO COMMODOR
    E + T
330 POKE 56334,PEEK(56334) AND
    254:POKE 1,PEEK(1) AND 251
340 FOR I=0 TO 7
350 L=53248+H*8+I:IF S=14 THEN
    L=L+2048
360 L=PEEK(L):A$(I)="........":
    L(I)=L
370 FOR N=0 TO 7:K=7-N
380 IF L>=(2↑K) THEN L=L-(2↑K):
    A$(I)=LEFT$(A$(I),N)+ER$+RI
    GHT$(A$(I),K)
390 NEXT:NEXT
400 POKE 1,PEEK(1) OR 4:POKE 56
    334,PEEK(56334) OR 1
410 PRINT"[CLEAR][DOWN][RIGHT][
    RVS]VIDEO (V) O CARTA (C)?"
420 PRINT"[RIGHT]──────────────
    ────────":REM SHIFT + C
430 GET A$
440 IF A$="C" THEN PRINT"[DOWN]
    [RIGHT]CARTA":OPEN 4,4:CMD
    4:B=14:GOTO 470
450 IF A$<>"V" THEN 430
460 PRINT"[DOWN][RIGHT]VIDEO[DO
    WN]"
470 PRINTCHR$(B);"[RIGHT]CODICE
     SCHERMO:";H:PRINT:PRINT
480 PRINTCHR$(15);"
    ────────";CHR$(8):REM TAST
    O COMMORORE + P
490 FOR I=0 TO 7:PRINTCHR$(15);
    "          |";A$(I);"|
    ";L(I);CHR$(8):NEXT
500 PRINTCHR$(15);"
    ────────":REM TASTO COMMOD
    ORE + Y
510 IF A$="C" THEN PRINT#4:CLOS
    E 4:POKE 198,0:WAIT 198,1:
    RUN
520 FOR I=1436 TO 1716 STEP 40:
    FOR G=0 TO 7
530 IF PEEK(I+G)=CS THEN POKE 5
    4272+G+I,0:GOTO 550
540 POKE 54272+I+G,15
550 NEXT:NEXT:POKE 198,0:WAIT 1
    98,1: RUN
560 END
```

# COMPUTER FEST '87

## mostra mercato dell'hardware software e tecnologie per la comunicazione e l'ufficio

Segreteria organizzativa: **PROMO EXPO** - Via Barberia, 22 - 40123 Bologna - Tel. (051) 33.36.57-33.27.42

# L'altra posta

*Il carteggio con i lettori continua; una breve rassegna sul contenuto delle vostre missive*

di Alessandro de Simone

E' mai possibile che dobbiamo ripetere tutte le volte che non rispondiamo MAI privatamente, nemmeno ai presidenti (scusa Ronald, ma vale anche per te), che non possiamo prendere in considerazione listati su carta (ma solo su disco), che è più economico telefonare (anzichè scrivere) soprattutto per coloro che abitano a Milano?

Il rischio che si corre è enorme: il cestino presente vicino alla mia scrivania ha dimensioni gigantesche e fagocita con piacere lettere, cartoline e subdoli omaggi e complimenti chiaramente inviati per secondi fini (esempio: "Voi, che siete candidati al premio Nobel per la miglior letteratura informatica del nostro Pianeta, potreste inviare a casa mia"... eccetera eccetera).

## SYStema di POKE

E veniamo alle notizie più succose dei nostri lettori che anelano acciocchè siano pubblicate.

Fabio Pellegrini, di Torino, ha scoperto la Sys del reset del C/128 (Sys 16348) e le seguenti Poke (sempre per C/128):
Poke 775,200 (lista solo i numeri di linea Basic)
Poke 775,169 (idem con patate)
Poke 808,100 (disabilita Restore)

Nonostante siano più che note, le ripubblico per la gioia dei neo-lettori.

Francesco Bizzini, di Catania, chiede: "Riuscite voi a far apparire graficamente la Luna ed i suoi crateri con accanto il Sole, in una sola riga di programma di C/128? Io sì!". Bravo, anche se, come affermi, ci sei riuscito per sbaglio (viva la sincerità).

*1 color 0,1: color 4,1: color 1,13: fast: graphic 1,1: for l=0 to 40: circle 1,160,100,l: next: color 1,8: circle 1,240,70,17: paint 1,240,70,1: slow*

Un anonimo, che non vuole essere scoperto nè cestinato, ci informa benevolmente sulla Sys 32800,123,45,6 (del C/128) che fa apparire il nome dei progettisti del C/128 stesso. Grazie, la conoscevo già e, come me, tanti altri lettori che posseggono il C/128 da molto tempo. Contento? Bè, ora ti cestino...

Restando ancora sul tema C/128, Stefano Tabacchiera chiede di digitare più volte di seguito una certa Sys e di commentare ciò che succede. Caro Stefano, non succede proprio niente: non potevi essere più chiaro nella tua richiesta di spiegazioni?

Adriano Locci, di Campo nell'Elba, si meraviglia che digitando un po' di Sys (a casaccio) il suo C/128 si blocca. Io mi meraviglierei del contrario! I messaggi "strani" che appaiono effettuando alcuni dei citati esperimenti non sono altro che quelli relativi al programma Monitor per lavorare in linguaggio macchina. Spiegare in dettaglio la sintassi (e l'uso) del Monitor richiederebbe alcune annate di CCC.

Ed ora un po' di Poke e Sys per C/64.

Cristiano Boaretto, di Monfalcone, suggerisce di provare le due strane Poke:

*Poke 22,0*

...e di seguito...

*Poke 22,1*

...e di spiegarne lo strano comportamento (ma preferisco rilanciare la domanda ai nostri lettori limitandomi a comunicare che la locazione 22 è riservata come puntatore per la stringa temporanea).

Inoltre ricorda che con Poke 56389,X si può modificare il lampeggio del cursore (va bè, lo sapevamo già).

Walter Quaglia invia un mini programma che contiene, al suo interno, alcune Poke e Sys che definisce interessanti:

```
10 poke 54240,6: rem=poke 53280,6
15 poke 54241,6: rem=poke 53281,6
20 poke 646,1: rem cursore bianco
30 sys 58692: rem cancella schermo
40 print"digita 10 caratteri";:sys43967x$:rem input x$
45 print x$
50 print"digita 39 caratteri"
70 sys 43899 y$:ify$= ""then70:rem get y$
80 print y$
90 k$=k$+y$:if len(k$)<39 then 70
100 fori=1 to 18:sys 59626:next:rem scroll up
110fori=1 to 24:printk$:next:rem riempe schermo
120 fori=1 to 24 step 3:poke781,i:sys59903:next:rem cancella una riga ogni tre
130 sys 58728:rem cursore home
140 sys 58592:rem attende pressione tasto run/stop
```

Dario Malfi, di Spinea (Ve), invia il seguente minilistato...

```
100 ti$="000000":for i=1 to 10000:next
110 print ti$
120 poke 53265,peek(53265)and239
130 ti$="000000":for i=1 to 10000:next
140 print ti$
160 poke 53265,peek(53265)or16
```

...assicurando che si possono risparmiare alcuni secondi durante l'elaborazione di calcoli lunghi e complessi; io però rimango un po' scettico al riguardo.

Raffaele Marzano, di Caivano (Na), (che non posso accontentare nella sua richiesta di individuare l'errore nel listato inviato) ritiene di aver trovato un sistema di protezione:

Se provate a registrare un programma assegnando il nome in cui è presente un carattere di virgola (,) come ad esempio...

Save "P,rova",8

...nella Directory apparirà con il semplice nome "P", ma tentando di caricarlo con Load "P",8 si otterrà un messaggio di "File not found error". E' ovvio che se si ricorre alla sintassi completa (Load "Prova",8) oppure al carattere di asterisco (*) il programma si carica egualmente. Se, però, nella directory è presente, prima di quel programma, un altro file il cui nome inizia con "P", non sarà possibile caricarlo. L'idea si presta, ovviamente, per nuove tecniche di protezione.

## Hardware

Fabrizio La Mura, Via V. Malcagni 134 Trani (Ba), ci tiene moltissimo a far sapere che non solo ha letto l'articolo pubblicato su CCC N. 34 ("Come fabbricare un mouse"), ma ha realizzato una tavoletta grafica migliorando il progetto presentato.

Il principio di funzionamento è lo stesso, afferma Fabrizio, e la differenza consiste nella diversa disposizione dei componenti. I potenziometri a slitta, stavolta, sono quattro, ma solo due sono collegati elettricamente. Meccanicamente, invece, troviamo due striscette rigide e trasparenti (mi sembra che il materiale sia il metametilmetacrilato, plexiglass per il volgo) percorse da una fessura, nel centro, larga abbastanza per farvi passare la punta di una matita.

Poichè le due fessure sono perpendicolari tra loro, ne consegue (elementare, Watson) che sarà disponibile un foro in cui imprigionare cinicamente la predetta punta di matita.

Dovrebbe esser chiaro, a questo punto, che gli altri due slider servono solo per garantire un migliore scorrimento della punta e delle fascette in metametilmetacrilato (plexiglass per i modesti).

Con software appropriato, assicura ancora Fabrizio, è possibile disegnare con enorme facilità soprattutto se, furbescamente, si deposita sul fondo della tavoletta un disegno da riprodurre sullo schermo in alta risoluzione.

Occhei, ed ora passiamo ad altro.

## Sprite su C/16 e Plus/4

Edoardo Bartolucci, di Fratterosa, afferma che il C/16, ed il fratello maggiore Plus/4, non devono sentirsi menomati rispetto al C/64 solo perchè non possono gestire Sprite. Ricorrendo alle istruzioni Gshape e Sshape è possibile memorizzare piccole aree di schermo e ricomporle ovunque. Invia anche un breve listato dimostrativo (che mi sono permesso di ridurre al minimo) per dimostrare quanto asserisce:

```
100 graphic 2,1
110 circle,10,10,5,5
120 sshapea$,1,1,20,20
130 scnclr
140 for a=0 to 100 step5
150 gshapea$,20+a,100,0
160 scnclr:next a
```

Naturalmente il nostro lettore ha (in parte) ragione; ma non posso fare a meno di esser cattivo: e con la velocità come la mettiamo? hai mai visto, sul C/64, la velocità alla quale può viaggiare un "vero" sprite?

Si potrebbe rispondere che, lavorando in L.M, la velocità di gestione potrebbe essere aumentata. Ma se andiamo con il L.M. sul C/16, bisognerebbe fare un confronto con ana-

HA
HB
HS
HI
HC
HT
Quota
HE
B
L
F7
F8
F2
F3
F4
F5
B A L C O N E
A SBALZO DAL TRAVE 22 - 23 A QUOTA +7.50
D I M E N S I O N I
L = cm 140
HI = cm 14
HC = cm 12
B = cm 30
HT = cm 50
HS = cm 24
HA = cm 5
HB = cm 8
HE = cm 28
Quota D'IMPOSTA SOLAIO = m +7.50
FERRI PER UNA STRISCIA LARGA 1 m
POSIZIONE F2
a
b
c
2fi 10 lung. cm 180
MISURE PARZIALI
a=cm 8 b=cm 164 c=cm 8
POSIZIONE F3
e
f
g
2fi 10 lung. cm 120
MISURE PARZIALI
e=cm 8 f=cm 100 g=cm 12
POSIZIONE F4
e
f
g
2fi 10 lung. cm 85
MISURE PARZIALI
e=cm 8 f=cm 65 g=cm 12

loghi programmi per C/64 e non so chi avrebbe la peggio....

Insomma, devo ricordare che, in certe occasioni, non basta che un computer "faccia" certe cose, ma occorre sapere, soprattutto, "in quanto tempo" le fa. Se, poi, gli sprite si usano soprattutto in giochi di destrezza, non so se mi spiego.

L'ingegner Franco Bonatta, di Bolzano, invia l'output di un programma (di cui è autore) a dir poco sconcertante: utilizzando il plotterino 1520 della Commodore riesce a svolgere in modo completo un intero progetto: disegni, calcolo e verifica di sezioni in cemento armato, solai, diagrammi e perfino relazione e computi metrici!

Il Bonatta si chiede, quindi, se il presunto "oscuramento" attuato nei confronti del gioiello 1520 sia dovuto alla necessità di vendere plotter e computer più grossi (e costosi).

Dubito che il motivo sia quello citato e non posso che ripetere la domanda: in quanto tempo il programma svolge un progetto completo? Fin dove si spinge la sua versatilità? Fino a che punto è sufficiente utilizzare l'output del plotterino così come è prodotto senza rimaneggiarlo per un reale utilizzo pratico?

Insomma: vale la pena utilizzare il programma oppure si fa più in fretta a pervenire agli stessi risultati con carta, penna e tecnigrafo?

In caso negativo, non possiamo fare a meno di sottolineare che con attrezzature modeste si possono fare "cose" modeste. Se, invece, il programma scritto dal nostro lettore serve davvero a far risparmiare tempo, bè, ne possiamo riparlare...

## Ditte fantasma

Alcuni lettori si sono meravigliati quando, rispondendo ad un'inserzione per saperne di più circa un drive recensito dal sottoscritto, hanno sempre e solo trovato come interlocutrice una... segreteria telefonica.

Sembra che il titolare della Ditta in questione abbia avuto, a campagna promozionale già iniziata, dei problemi per la cessione dei locali in cui avrebbe dovuto insediarsi. Poichè, appunto, la campagna pubblicitaria non poteva esser procrastinata, l'unica soluzione possibile è risultata quella di ricorrere ad un recapito telefonico che, come tale, non poteva che risultare "di fortuna".

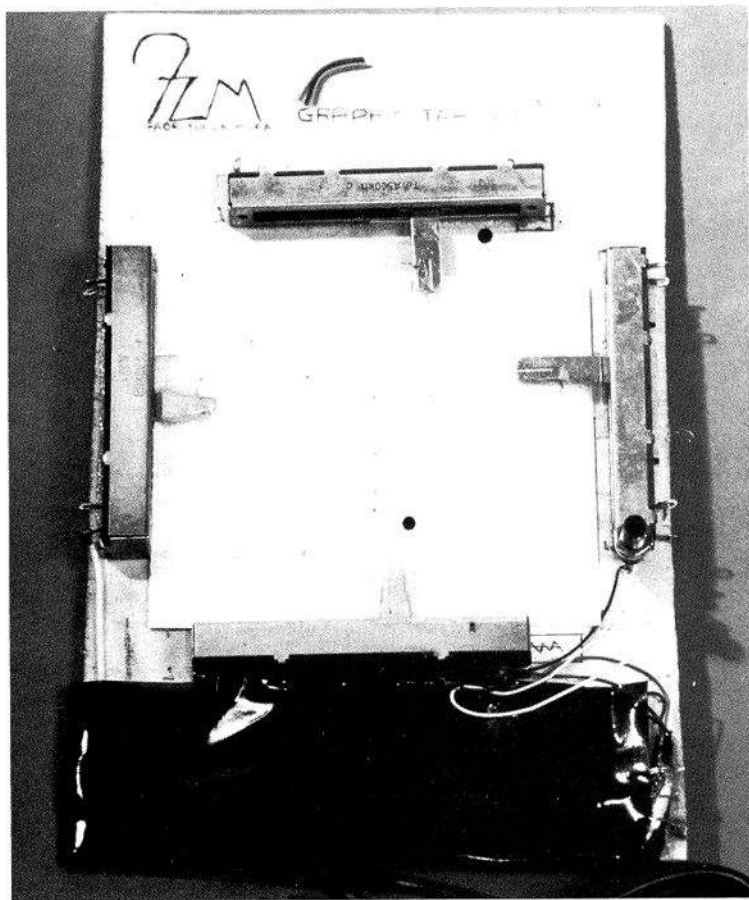
Mi sento, comunque, in dovere di affermare che nessun lettore, che abbia acquistato il drive dalla Ditta citata, abbia avuto a che lamentarsi; ai più sospettosi si potrebbe suggerire di ordinare il drive per contrassegno.

Gli irriducibili si limiteranno a... non acquistarlo(!), nonostante la cifra proposta per il prodotto sia la più bassa (per quel che mi risulta) praticata in Italia.

Altri, poi, pensano che il prezzo sia troppo basso per un disk drive: siamo alle solite! Ma lo sapete che, ormai, un drive da 3.5 (capace di 800 K) può costare meno di 400 mila lire?

Con la tecnologia attuale è possibile realizzare apparecchi di una qualità tale che, qualche anno fa, non era neppure ipotizzabile.

A Pierluigi Baglioni, che invia un diagramma di flusso con preghiera di trasformarlo in programma, non posso che suggerire di rileggere l'inserto del N.39, interamente dedicato alla corretta gestione dei file sequenziali.

Al dottor Carlo Bondioli devo purtroppo rispondere che sarà difficilissimo trovare sul mercato programmi affidabili per la gestione di cartelle cliniche, buste paghe e gestione affitti. In ogni caso dubito che il software richiesto, se trovato, possa girare anche su cassetta.

# Amiga nel paese del CLI

*Siete in grado di indicare un computer dotato di un sistema operativo più potente dell'Amiga?*

di Luigi Callegari

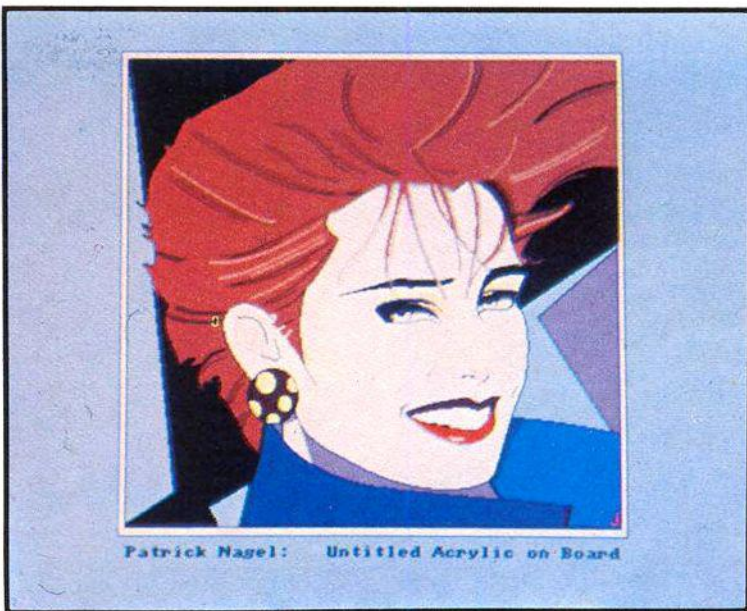
Come molti sanno, esistono due versioni di AmigaDOS: la 1.1 e la 1.2. Questa ultima, oltre a fornire comandi aggiuntivi e ad avere meno bug, dispone di un migliore sistema (algoritmo) per la formattazione dei dischetti. Per trasformare un dischetto inizializzato con la versione 1.1, in uno più veloce, è necessario usare il comando DISKCOPY.

Apriamo una doverosa parentesi: taluni Workbench non hanno questo comando nella directory "C" e quindi non è direttamente eseguibile da CLI (i cui comandi disponibili, ricordiamo, sono solo quelli specificati in questa directory); a meno che non si esegua manualmente da CLI, se non è stato previsto già nel file di inizializzazione "S/Startup-Sequence" un comando del tipo...

PATH SYSTEM ADD (R)

...che "inserisce" anche i comandi presenti nella directory "System", ove risiede talvolta il comando DISKCOPY (oltre a SETMAP, FORMAT/INITIALIZE e qualche altro) tra quelli gli eseguibili da CLI.

Tornando a Diskcopy, questo esegue una copia integrale di un dischetto, usando uno o due drive, creandone uno nuovo formattato automaticamente con il nuovo sistema "1.2" migliorato. Il formato è:

DISKCOPY (FROM) drivesorgente TO drivedestinaz (NAME) (nomevolume)

Il "drivesorgente" è il drive (DF0: DF1: eccetera) ove vogliamo inserire il dischetto originale da duplicare (anche se in vecchia versione DOS); il "drivedestinaz" è quello ove si vuole inserire il dischetto copia. La parola FROM è opzionale, TO è obbligatoria. Opzionalmente si può specificare il nome da assegnare al volume copia, specificando NAME seguito dal nuovo nome del disco; tralasciando questi parametri, il disco copia avrà lo stesso nome del disco originale. Si noti che però il sistema è in grado di distinguerli per la data e l'ora di formattazione incisa automaticamente, anche se hanno nome identico. Il disco copia può essere non formattatto. Coloro i quali non posseggono drive supplementari debbono usare un comando del tipo:

DISKCOPY df0: TO df0: (Return)

Ovviamente il sistema provvede a richiedere, volta per volta, l'estrazione e l'inserimento dei dischetti. In condizioni standard sono necessari tre scambi. I possessori di due drive possono usare qualcosa come:

DISKCOPY df1: to df0: NAME "back-up" (R)

In questo caso verrebbe copiato il contenuto del dischetto, sistemato nel primo drive esterno, nel dischetto contenuto nel drive interno ed assegnando a questo il nuovo nome "back-up". Attenzione: nel dischetto di copia verranno perduti tutti i vecchi dati, sostituiti da quelli contenuti nel dischetto da copiare!

Nel caso in cui il nome scelto per un disco directory-file non ci piaccia, esiste il comando seguente (le parentesi, come sempre nelle nostre spiegazioni, indicano parametri o parole opzionali):

RENAME (FROM) nomevecchio (TO o AS) nomenuovo

La parola FROM è opzionale solo quando i parametri sono specificati nell'ordine corretto (prima il nome vecchio, poi il nuovo). TO oppure AS sono equivalenti ed opzionali, ma se specificati debbono essere necessariamente seguiti dal nuovo nominativo desiderato. Esempio:

RENAME mary AS lory (R)

Cambia il nome del file "mary", prelevato dalla directory corrente, in "lory". Questo comando, comunque, non consente di spostare file e directory molto agevolmente nella struttura ad albero di AmigaDOS, in quanto il nome nuovo del file non può riferirsi ad una directory diversa da quella ove è contenuto il file stesso.

Ciò significa, ad esempio, che tentando di spostare il file "Gigi" (contenuto nella directory "uomini") verso la directory "donne" usando qualcosa come...

RENAME uomini/Gigi AS donne/Gigi (R)

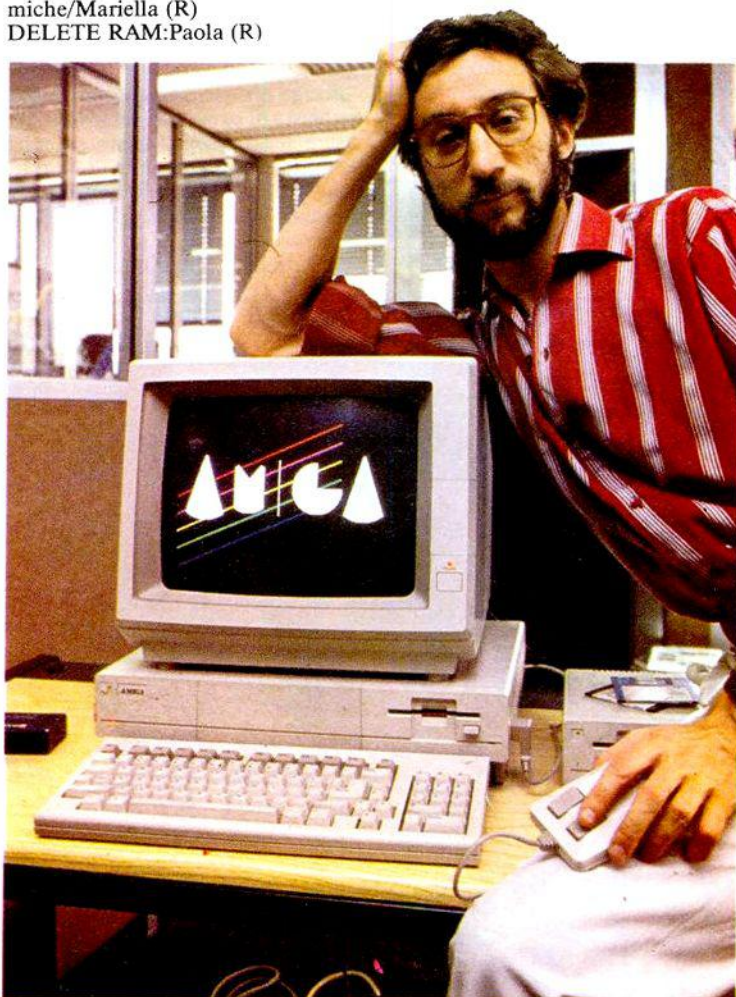
...non si otterrà, come alcuni astuti potrebbero avere pensato avendo appreso le convenzioni dei nomi in Amigados, lo spostamento del file nella directory "amiche", ma un messaggio di errore. Per ottenere spostamenti di file nella struttura ad "albero" di AmigaDOS si deve usare il comando COPY, che permette anche la ridenominazione del file interessato.

Ad esempio: per spostare il file "Paola" presente nel RAMdisk nella sub-directory "amiche" della directory "donne" presente nel dischetto del drive interno, cambiandogli il nome in "Mariella" userò:

COPY RAM:Paola TO df0:donne/amiche/Mariella (R)
DELETE RAM:Paola (R)

Questo perchè il file "Paola" viene inserito al posto del file "Mariella" nella sub-directory "amiche"; se non esiste questo nominativo ("Mariella") viene automaticamente creato un file con questo nome, ove viene inserito il contenuto del file "Paola"; Se il file "Mariella" dovesse già esistere nella sub-directory "amiche" allora il suo contenuto verrebbe rimpiazzato dal contenuto del file "Paola". Vedete dunque come sia possibile, usando COPY, spostare e ridenominare un file in un colpo solo.

Ricordate bene questa regola generale di AmigaDOS: un file ove scrive-

re, se non esiste già viene creato, se invece esiste viene utilizzato e cancellato dal nuovo contenuto. Tale regola vale anche nei programmi applicativi, sotto BASIC (se non si specifica APPEND nei file sequenziali) ed in altri linguaggi che usano regolarmente AmigaDOS.

Altro comando molto utile è JOIN, che permette di fondere insieme sino a quindici file in uno solo. Il formato è:

JOIN nome1 nome2 .... nome15 AS nomedest

Gli argomenti "nome1","nome2" sono nomi di file, con le consuete convenzioni AmigaDOS. "nomedest" è il file ove viene inserito il risultato del concatenamento dei file. Non può essere uno dei file di argomento e, come al solito, se non esiste viene creato mentre se esiste già il suo contenuto è sostituito dal risultato dell'operazione di fusione.

JOIN ram:amiga1 ram:amiga2 amiga3 AS df0:archivio/Amiga (R)

inserisce nel file "Amiga" presente (o creato) nella directory "archivio" (che deve esistere) nel dischetto nel drive interno, ordinatamente: il file "amiga1" prelevato dalla root directory del ramdisk, il file "amiga2" prelevato sempre dalla directory principale del ramdisk, il file "amiga3" prelevato dalla directory corrente (CD).

Sotto AmigaDOS esistono alcuni comandi "di bellezza", ovvero più di scena che utili. Uno è il seguente:

PROMPT stringa

Ridefinisce la stringa usata dal sistema per segnalare di essere pronto a ricevere un nuovo comando. Solitamente si tratta del numero della finestra seguito da un simbolo di "maggiore" (1>). Usando PROMPT la "stringa diviene il nuovo simbolo di richiesta di comando. Se si vogliono usare spazi bisogna usare gli apici. Per indicare il numero di finestra nella stringa si usa il simbolo di "percentuale" (%). Esempio:

PROMPT "% dimmi tutto:" (R)

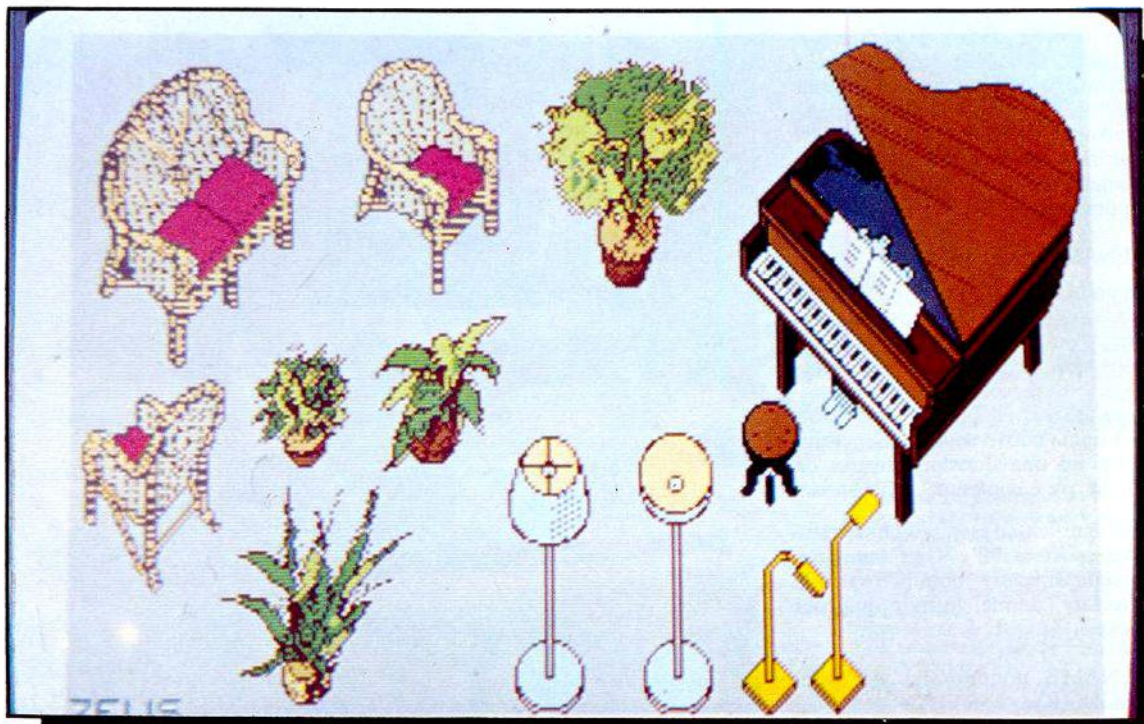
In codesto caso ad ogni richiesta di comando il sistema visualizzerà ogni volta il numero di task (1, 2, 3...) seguito da "dimmi tutto:".

## Operando in multitasking

Si è detto mille volte che il sistema operativo di Amiga è il primo per personal computer ad usare un vero multitasking. Una possibile applicazione è l'uso di più finestre CLI operanti insieme. Diviene ad esempio possibile usare l'editore per redigere un file in una finestra, mentre in un'altra gestiamo il riordino di un dischetto e in un'altra finestra CLI ci prepariamo ad inviare dati alla stampante. Esistono vari comandi per ottenere un funzionamento "concorrente" di questo tipo. Incominciamo con:

NEWCLI (CON: xpos, ypos, larg, altz, titolo)

Questo comando permette di aprire una nuova finestra di CLI, con di-

mensioni e posizione opzionalmente specificabili, e di passarvi il controllo (ovvero digitarvi comandi). Come al solito, per rendere attiva una finestra differente da quella corrente, bisogna posizionarvi la freccia del cursore e clickare col mouse (pulsante sinistro). NEWCLI, da solo, apre una finestra standard con valori di default.

Specificando invece CON: è possibile fissare misure e posizione:

"xpos" è la posizione dello spigolo superiore sinistro della nuova finestra, espressa come un numero di pixel di distanza dal bordo superiore sinistro dello schermo. In difetto (default) vale zero.

"ypos" è la posizione verticale dello spigolo superiore sinistro della nuova finestra, espressa in modo simile a "xpos". In difetto vale zero.

"larg" ed "altz" sono obbligatoriamente da specificare quando si è specificato CON:. Indicano rispettivamente la larghezza e l'altezza in pixel della nuova finestra. Il valore minimo è 90x25 ed il valore massimo, in alta risoluzione interlacciata, è 640x200.
"titolo" è il titolo, opzionale, da assegnare alla nuova finestra. Nel caso lo si specifichi ddevono essere stati specificati anche tutti i precedenti parametri. Esempi:

WINDOW ,,300,120 (R)
WINDOW 20,20,120,120,"Ciccio" (R)

Il primo comando apre, e rende operativa, una nuova finestra di CLI, priva di titolo, partendo dallo spigolo superiore sinistro del video, larga 300 punti ed alta 120. Il secondo comando apre ed attiva una finestra quadrata, con lato di 120 pixel, in coordinate (20, 20) rispetto allo spigolo superiore sinistro dello schermo, con nome "Ciccio". Queste finestre possono essere disattivate digitandovi, all'interno, il comando classico:

ENDCLI (R)

Un comando nuovo, presente solo nella versione 1.2 del sistema operativo, permette di fissare la priorità, ovvero la "importanza", di ogni finestra CLI in modo che si possa favorire e rendere più efficienti i processi più importanti. Ad esempio, eseguendo contemporaneamente la codificazione di un testo sorgente con un compilatore e l'edizione di uno nuovo con un editor, è certamente preferibile rendere "più importante" agli occhi del sistema l'editor; se rallenta troppo, infatti, la digitazione diventerebbe critica. L'esecuzione contemporanea di più programmi, come è intuitivo, comporta sempre un certo rallentamento: ricordiamo che, di microprocessori, Amiga ne ha uno solo, anche se è il potentissimo 68000 supportato dai vari co-processori; si trova quindi costretto a lavorare in una sofisticata "divisione di tempo" ("time-sharing") per ogni finestra CLI attivata ("task").

Il comando che permette di minimizzare questi effetti è:

CHANGETASKPRI priorit

Il parametro numerico "priorit" può essere un numero qualunque: bisogna quindi essere prudenti nell'assegnare i parametri perchè il sistema non effettua controlli; ne consegue che risulta possibile, purtroppo, assegnare valori troppo bassi o troppo alti, per cui il sistema diventa ingovernabile se gli viene ordinato di privilegiare un certo task, trascurando, magari, di scandire la tastiera o di muovere il cursore del mouse.
Altro comando fondamentale per il multitasking ha il formato:

RUN task (+ task)(+ task)...

Permette di lanciare contemporaneamente più programmi, mantenendo la possibilità di inserire altri comandi nella corrente finestra CLI. Ad esempio:

RUN notepad + clock + format drive df1:(R)

Questo comando lancia il blocco note, l'orologio e il formattatore per il dischetto sistemato nel primo drive esterno; solo in seguito il cursore risulta pronto a ricevere nuovi comandi. Quando si esegue, ad esempio, un semplice...

CLOCK (R)

...viene egualmente aperta una nuova finestra CLI (task) ma il sistema esegue solo quello. Invece, con RUN, è possibile eseguire contemporaneamente tutti i programmi ed i comandi che si vogliono, compatibilmente con la capacità della memoria.

Per la gestione di un serio multitasking di vari programmi applicativi, è necessario espandere ad almeno 1 mega la memoria RAM del nostro gioiello. Sappiamo benissimo che questo fatto può esser considerato un inconveniente, ma se qualche Atarista invidioso vi prende in giro, potete sostenere che il suo pur ottimo computer non potrebbe eseguire contemporaneamente la gestione di un word-processor, di un tabellone elettronico, di una stampante e di un modem nemmeno se lo espandesse a 8 megabyte!

# Chiedo la parola!

## Una versione migliorata del listato apparso sul N.41 della nostra Rivista

di Gianluca Torta

Leggendo un precedente articolo, e soprattutto i suggerimenti della Redazione posti al termine dello stesso, è nata l'idea di potenziare il già ottimo programma ("Alla ricerca della parola", di M. Maggi).

Il programma in Basic e L.M, di cui stiamo parlando, consentiva ricerche di tipo statistico su file di testo memorizzati con il popolare word processor Easy Script.

La nuova versione riportata in queste pagine pone un rimedio, nel modo più semplice possibile, a tutte le "carenze" citate.

Anzitutto è stato necessario creare due routine in L.M.: la prima, allocata da C000 a C038 permette il caricamento del file di testo; la seconda (C039-C0BF) analizza il file.

**SCHEDA TECNICA**

Software applicativo di utilità per la gestione di file generati dal word processor "Easy Script".

Idoneo per computer C/64 e difficilmente adattabile ad altri computer Commodore.

Ideale si rivela l'uso del disk drive, che consente di sfruttare al massimo le sue caratteristiche.

Consigliata la rilettura del fascicolo C.C.C. N.41.

Anche il programma pubblicato in queste pagine è contenuto nel disco "Directory" di questo mese.

Il ricorso al codice macchina si è reso indispensabile dal momento che un'analisi in Basic avrebbe richiesto... secoli.

Si noterà che il listato Basic è stato di molto esteso rispetto alla versione del N.41 allo scopo di aumentarne la chiarezza e (scusate l'immodestia) la professionalità.

### Il programma

Dopo aver scelto la periferica e la possibilità offerta per la distinzione tra caratteri maiuscoli e minuscoli (o meno), appare il menu principale; le operazioni di analisi per parola e di utilizzo del vocabolario non sono accessibili fino a che non si è caricato in memoria il file da analizzare (premendo il tasto F).

Chi possiede il solo datassette non potrà utilizzare il vocabolario a causa dei caricamenti interminabili ma, soprattutto, per la impossibilità di legare, mediante append, più file sequenziali tra di loro.

Il vocabolario è gestito in Basic; non valeva la pena, infatti, scomodare il L.M. per il modesto risparmio di tempo che si sarebbe ottenuto. Vi si possono introdurre parole in modo diretto, oppure dopo la ricerca di una data parola.

Il programma, naturalmente, confronta tutte le parole contenute nel vocabolario presente su disco (che si crea un po' per volta) con il file presente in memoria Ram.

E' stata aggiunta, infine, l'opzione per "parole" di 255 caratteri! Si ricorda che è possibile la creazione di più vocabolari diversi sullo stesso disco.

### Nota della Redazione

A quanto pare i nostri suggerimenti, posti spesso al termine degli articoli, sono presi in seria considerazione dai nostri lettori. Ringraziamo, pertanto, il simpatico Gianluca Torta di Cuneo che sarà premiato con una manciata di banconote da 10.000 lire (e spiccioli vari...) per il contributo apportato.

Agli altri non rimane che rinnovare il nostro invito a scrivere nuovi listati oppure a migliorare i programmi che pubblichiamo.

Il solito avvertimento è d'obbligo: non inviate programmi prima di aver telefonato in Redazione (02.84.67.348) per stabilire se il lavoro è interessante o meno.

```
10 rem ***********************
   ********
20 rem alla ricerca della paro
   la v 1.2
30 rem                    ***
40 rem scritto da torta gianlu
   ca  (cn)
50 rem ***********************
   ********
60 :
68 poke 53280,0:poke 53281,0:p
   oke 646,1:print chr$(14)chr
   $(8):gosub 6000
69 print"[clear][down][right](
   R)egistratore"
70 print"[down][right](D)isk d
   rive"
71 print"[down][right](E)sce"
72 geta$:ifa$="" then 72
73 if a$="e" then print"[clear
   ]":end
74 if a$="r" then per=1:goto 7
   7
75 if a$="d" then per=8:goto 7
   7
76 goto 72
77 print"[clear][down][right](
   D)istinzione mai/min"
78 print"[down][right](N)essun
   a distinzione"
79 get a$:if a$="" then 79
80 if a$="d" then qq=0:goto 85
81 if a$="n" then qq=1:goto 85
82 goto 79
85 print"[clear][down][right](
   F)ile da analizzare"
86 if per=1 then 106
90 print"[down][right](C)reazi
   one vocabolario"
100 print"[down][right](I)intro
    duzione dati in vocabolario
    "
105 print"[down][right](U)tiliz
    zo del vocabolario"
106 print"[down][right](A)nalis
    i per parola"
107 print"[down][right](R)ipart
    e"
110 get a$:if a$="" then 110
120 if a$="f" then goto 160
121 if a$="r" then run 69
122 if per=1 and sw=1 then 140
123 if a$="i" then gosub 4000:w
    $=uu$:gosub 5000:goto 85
124 if a$="c" then ww$="w":gosu
    b 1000:close2:close1:goto 8
    5
125 if sw<>1 then 110
130 if a$="u" then goto 300
140 if a$="a" then goto 240
150 goto 110
160 input"[clear][down][right]N
    ome file (max 16)";nf$
165 if len(nf$)>16 then 160
167 yy$=nf$:ind=49479:gosub 200
    0:poke 49185,195
170 poke 49153,len(nf$):poke 49
    164,per:poke 49166,int(per/
    2):sys(49152)
180 open1,8,15:gosub20000:close
    1:if dix=1thenpoke peek(491
    85)*256+81+peek(782),0
190 sw=dix:goto 85
240 gosub 4000:w$=uu$
250 yy$=w$:ind=49499:gosub 2000
260 poke 251,len(w$)
270 print"[clear]":sys (49209)
279 zz$=w$:gosub 3000:print:pri
    nt"premi un tasto":poke 198
    ,0
280 getpo$:ifpo$=""then 280
281 if per=1 then 287
282 print"[clear][down]Memorizz
    o la parola in vocab.(y/n)"
283 getpo$:ifpo$=""then 283
284 ifpo$="y" then gosub 5000:g
    oto 287
285 if po$="n" then goto 287
286 goto 282
287 goto 85
300 ww$="r":ww=0:gosub 1000:if
    dix<>1 then 370
321 if st<>0 then 360
322 input#2,a$
325 input#2,a:poke 251,a
330 yy$=a$:ind=49499:gosub 2000
340 print"[clear][right]":poke
    49241,195:sys (49209)
350 zz$=a$:gosub 3000
351 print:print"premi un tasto"
    :poke198,0
```

```
 355 get po$:if po$="" then 355
 356 a$="":goto 321
 360 print"[down]o.k. vocabolari
     o analizzato!":for a=1 to 2
     000:next
 370 close2:close1:goto 85
 500 data 169,0,162,72,160,193,3
     2,189,255,169,8,162,8,160,8
     ,32,186,255,32,192
 510 data 255,162,8,32,198,255,1
     60,0,32,207,255,153,80,195,
     32,183,255,208,9,200
 520 data 208,3,238,33,192,76,28
     ,192,169,8,32,195,255,32,20
     4,255,96,169,0,133
 530 data252,133,253,162,0,142,2
     55,207,173,89,192,141,254,2
     07,141,84,192,141
 535 data 127,192,160,0,189
 540 data80,195,133,254,189,80,1
     95,201,0,240,97,217,92,193,
     240,5,217,91,194,208,51
 550 data200,232,208,3,238,89,19
     2,196,251,208,228,169,1,133
     ,199,172,255,207
 560 data142,255,207,185,80,195,
     32,210,255,200,208,3,238,12
     7,192,204,255,207,208
 570 data239,230,252,208,2,230,2
     53,169,0,133,199,76,65,192,
     174,255,207,173,254,207
 580 data 141,89,192,165,254,32.
     210,255,165,197,201,63,240,
     16,173,141,2,201,1
 590 data 240,249,232,208,3,238,
     89,192,76,65,192,96
1000 rem predispone file
1010 input"[clear][down]Nome del
      voc (max 16)";v$
1012 if len(v$)>16 or len(v$)<1
     then 1010
1015 if per=1 then open 1,1,ww,v
     $:goto 1030 .
1020 open 1,8,15:open 2,8,2,v$+"
     ,seq,"+ww$:gosub 20000
1040 return
2000 rem 'pokeizza' yy$ da ind i
     n poi
2005 for i=0 to 1
2010 for j=1 to len(yy$)
2015 cc$=mid$(yy$,j,1):sg=-sgn(a
     sc(cc$)/128-1)
2017 if (cc$<"a" or cc$>"z") and
      (cc$<"A" and cc$>"Z") then
      sg=0
2020 poke ind+j+i*255,asc(cc$)+i
     *qq*sg*128
2030 next j:next i:return
3000 rem stampa risultato
3010 print:print:print"il numero
      di volte che [rvs]"zz$"[rv
     off] compare e' "peek(252)+
     256*peek(253)
3020 return
4000 rem introduzione parola
4010 uu$="":print"[clear][down][
     right]Introduci parola desi
     derata(max 255)":poke 650,1
     28
4020 print"[right]batti /return/
      quando finisci":print"[dow
     n][right]";
4030 geta$:ifa$="" then 4030
4040 if a$=chr$(13) and uu$<>""
     then 4060
4050 print a$;:uu$=uu$+a$:if len
     (uu$)<255 then 4030
4060 print"[rvs]<[rvoff]":poke 4
     9241,195:return
5000 rem appende nuova parola a
     voc
5010 ww$="a":gosub 1000:if dix<>
     1 then 5040
5020 print#2,w$
5030 print#2,len(w$)
5040 close 2:close 1:return
6000 rem carica il codice macchi
     na
6010 restore:bb=0:for j=49152 to
      49343
6020 read a:bb=bb+a:poke j,a
6030 next j
6040 if bb<>29419 then print"[cl
     ear][down][right]Errore nei
      data!":end
6050 return
20000 dix=1:input#1,a,b$,c,d:open
      3,8,3:close3
20005 if a<>0 then print"[clear][
     down]ATTENZIONE!: "b$:for r
     i=0 to 2000:next:dix=0
20010 return
```

# Ed è subito Smau

*Rivelate, nel corso di una cordiale intervista, le novità Commodore che vedrete al Sim ed allo Smau; ed anche quelle che non vedrete...*

**di Alessandro de Simone**

Abbiamo avuto modo di dissertare, proprio nel numero scorso, sulla professionalità dimostrata dal nuovo staff della Commodore Italiana.

Una nuova occasione per sottolineare la sensibilità dei nuovi dirigenti nei confronti della stampa specializzata (e quindi, in ultima analisi, degli utenti finali) è stata offerta dalla immediata disponibilità del Direttore Supporti, Paolo Morganti, a rilasciare un'intervista per il nostro periodico, protrattasi, negli uffici della sede centrale, ben oltre il limite di tempo concordato.

Gli argomenti affrontati sono stati numerosissimi, tanto da esser costretti a riassumerli in queste colonne.

## Il futuro dei "piccoli"

Il C/64 sarà ancora prodotto per molto tempo ancora e sostenuto con varie iniziative.

La proposta Telematica, di cui ci siamo occupati diffusamente, ha fornito la risposta che si attendeva con fiducia e gli sviluppi si avvertiranno tra brevissimo tempo. La disponibilità di un modem Commodore "ufficiale", per giunta omologato dalla SIP, offrirà varie opportunità sia agli utenti stessi che a numerose ditte di servizi che, considerando il successo di vendita del modem, provvederanno quanto prima a sfruttare intensivamente il prezioso accessorio.

Aspettatevi, quindi, di assistere alla proliferazione di banche dati, di posta elettronica e di tutti quei servizi che, grazie al C/64, diventeranno una concreta realtà anche in Italia.

Il C/128 sarà ancora prodotto nella sua versione più economica, mentre la versione "D" (con il drive incorporato) avrà una produzione più limitata a causa dello scarso favore incontrato; è un vero peccato perchè, a mio parere, il C/128D (che uso abitualmente) è una bella macchina e meritava un migliore destino.

Del C/16 e del Plus/4, ovviamente, non si parla più, mentre verranno proposte diverse periferiche interessanti.

## Le periferiche

Nel campo delle memorie di massa certamente verrà osservato con attenzione il drive da 3.5 che, tiene a precisare Paolo Morganti, verrà introdotto per andare incontro alle esigenze dell'utenza che vede, nel nuovo media, una concezione più moderna di un sistema computerizzato.

Naturalmente vi saranno problemi di compatibilità con il software già disponibile, non tanto per l'ovvia impossibilità di introdurre dischetti di formato diverso, quanto per la diversa formattazione dei dischi stessi.

Soprattutto i programmi protetti, che fondano la propria inviolabilità su elaborazioni di tracce e settori del 1541, non potranno funzionare al 100% se "riversati" su dischetto da 3.5.

Va da sè che i programmi professionali che richiedono esplicitamente il 1541, dovranno necessariamente funzionare con tale periferica, mentre non è esclusa la proposta di software specifico per il nuovo drive. Anzi, dal momento che questo sarà, almeno nei primi tempi, relativamente sconosciuto, è probabile che possa contribuire a limitare il fenomeno della pirateria, almeno per ciò che riguarda il software diffuso nel nuovo formato.

E' bene sottolineare, però, che sarà possibile riversare su disco qualsiasi altro programma non protetto; la "trasparenza" del drive è infatti totale e l'utente non dovrà preoccuparsi di apportare alcuna variazione ai programmi in suo possesso.

La notevole disponibilità di blocchi liberi per ciascun disco, quindi, aprirà nuove possibilità per la gestione di archivi di grosse dimensioni, oppure, semplicemente, per conser-

vare su un solo disco tutto ciò che, prima, richiedeva più di un supporto; e tutto ciò oltre alle indiscutibili comodità offerte dal sistema 3.5.

Rimane il fatto che il nuovo drive verrà considerato con attenzione quasi esclusivamente da coloro che già posseggono un 1541 e che intendono dotare il proprio sistema di un secondo drive. Il prezzo? Praticamente identico a quello del tradizionale 1541!

Per ciò che riguarda le stampanti vi saranno diverse novità che potranno essere ammirate presso lo stand Commodore in occasione dello Smau. Possiamo anticipare soltanto che il prezzo sarà piuttosto contenuto e sicuramente in linea con la qualità offerta (superiore, comunque, alla MPS/803). Anche per l'Amiga saranno disponibili stampanti decisamente convenienti, pur se rivolte ad un'utenza professionale.

I monitor avranno linea moderna ed elettronica "ridisegnata" allo scopo di offrire standard qualitativi adatti alla riproduzione di package grafici sofisticati quanto si vuole.

## Ciò che non si vede

Oltre che allo Smau, la Commodore sarà anche presente al Sim con uno stand che, più che a mostrare il visibile, mostrerà... l'invisibile. Con ciò vogliamo riferirci al settore dei servizi e all'assistenza post-vendita in particolare.

Sorgeranno in Italia, infatti, numerosi punti di vendita, dotati di marchio inconfondibile, specializzati in articoli Commodore. Tale organizzazione commerciale renderà disponibile, di fatto, personale qualificato al servizio dell'acquirente; si tende, in altre parole, ad evitare che l'utente si affidi a negozi non specializzati dietro il cui bancone c'è qualche commesso che, dotato di sola buona volontà, suggerisce una marca o un modello.

Soprattutto ora, con la serie Amiga, si è avvertita l'esigenza di mettere a disposizione dell'utente punti di vendita specifici per consigliare la configurazione ottimale per le esigenze più varie.

La stessa organizzazione, estesa in tutt'Italia, eviterà disparità eccessive di prezzi che oscilleranno di pochissimo intorno al prezzo ufficiale dettato dalla Commodore Italia.

Dei centri di assistenza e riparazioni, di cui ci siamo occupati nel numero scorso, se ne riparlerà diffusamente anche allo Smau e si continuerà nella selezione, severissima, di altri laboratori di elettronica che vorranno offrire la propria disponibilità all'espletamento del delicato servizio.

## Il settore professionale

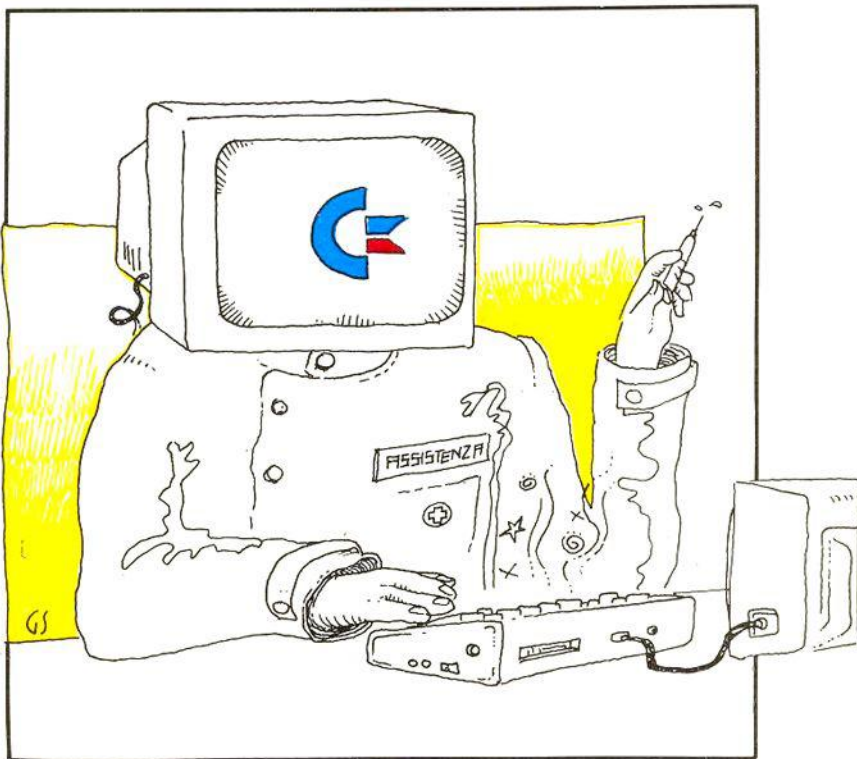
Per ciò che riguarda la serie Amiga, sulla quale, inutile dirlo, la Commodore punta moltissimo, verrà istituito un vero e proprio centro di commercializzazione del software la cui edizione dovrebbe essere in larga parte in lingua italiana.

Anche il settore dei compatibili non sarà trascurato, soprattutto alla luce delle unità vendute che, nel corso dell'86 e del primo semestre 87, ha raggiunto cifre più che soddisfacenti.

Tra breve vi sarà un autentico boom di pubblicazioni tecniche specifiche per Amiga, pur se non ufficiali Commodore.

## Allo stand Commodore

Interessanti, quindi, risulteranno gli stand Commodore del Sim e dello Smau, per coloro che avranno la fortuna di visitarli.

E' prevista un'area appositamente dedicata ai videogame, ed isolata dallo stand vero e proprio, per selezionare opportunamente le due categorie di visitatori: quelli che il computer l'hanno già (e che si informano, quindi, sugli accessori da procurarsi per espanderlo) e coloro che, invece, lo sognano ancora e che vogliono toccare con mano (o meglio, con joystick) l'indiscussa superiorità dei prodotti Commodore.

# La Grande Libreria Systems

Autori Vari

**64 Programmi per Commodore 64**

Giochi, grafica, gestione delle stringhe, musica, numeri, gestionali.

**Lire 4.800**

Autori Vari

**I miei amici C16 & Plus4**

Un manuale pratico per padroneggiare il basic di questi computer.

**Lire 7.000**

Autori Vari

**Strategie vincenti per Commodore 64**

Le strategie per tutti i classici del videogioco: per giocarli, vincerli o programmarli.

**Lire 5.800**

Autori Vari

**62 Programmi per il Vic 20, C16 e Plus 4**

Giochi, grafica e routine per imparare a programmare.

**6.500**

Roberto Didoni, Guido Grassi

**Utilities e giochi didattici**

Raccolta di programmi pratici per tutti i Commodore e lo Spectrum.

**Lire 6.500**

Giovanni Mellina

**Tutti i segreti dello Spectrum**

4 passi nella Rom: come usare le più importanti routine del sistema operativo.

**Lire 7.000**

Roberto Didoni, Guido Grassi

**Simulazioni e test per la didattica**

Teoria e listati per Vic 20, C16, C64 C128 e Spectrum Sinclair.

**Lire 7.000**

Paolo Goglio

**Impara giocando il basic dello Spectrum**

Esercizi pratici per entrare nel vivo della programmazione.

**Lire 7.000**

Clizio Merli

**µPascal per Commodore 64/128**

Un manuale completo per il programma compilatore

**Lire 7.000**

Umberto Colapicchioni e Luca Galuzzi

**Dal registratore al drive del C64**

Tutti i segreti delle memorie di massa del Commodore 64

**Lire 7.000**

Autori Vari

**ADA**

Il linguaggio passepartout dei computer degli anni '80.

**Lire 5.000**

Clizio Merli

**Il linguaggio PASCAL**

Un manuale tascabile per lo studio e la programmazione.

**Lire 5.000**

---

Sì, voglio arricchire la mia biblioteca con i seguenti volumi al prezzo di copertina + lire 3.000 per spese di spedizione.

- ☐ **64 Programmi per Commodore 64**
- ☐ **Strategie vincenti per i tuoi videogames**
- ☐ **62 Programmi per Vic 20 C16 e Plus77**
- ☐ **Utilities e giochi didattici**
- ☐ **Tutti i segreti dello Spectrum**
- ☐ **Simulazioni e test per la didattica**
- ☐ **Imparare giocando il basic dello Spectrum**
- ☐ **I miei amici C16 e Plus4**
- ☐ **Pascal per Commodore 128**
- ☐ **Dal registrattore al drive del C64**
- ☐ **ADA**
- ☐ **Il linguaggio Pascal**

**Nome** ..........

**via** .......... **N.ro.** .......... **telefono** ..........

**CAP** .......... **Città** ..........

Su tale importo mi praticherete lo sconto del 10% in quanto abbonato a ☐ *Commodore Computer Club* ☐ *Personal Computer* ☐ *Com puter* ☐ *VR Videoregistrare.* Pertanto vi invio la somma soltanto di lire ..........

Valore dell'ordine lire..........

Ritagliare e spedire in busta chiusa regolarmente affrancata a Systems Editoriale - V.le Famagosta, 75 - 20142 Milano.

# Supertape C/16

*Una nuova versione per realizzare caricamenti più rapidi con il registratore*

di Sante Mercurio

Il programma che vi apprestate ad esaminare è un sistema pratico, rapido e semplice che permette il salvataggio e il caricamento di programmi in tempi piuttosto ristretti.

Dopo aver caricato, e attivato, il programma di queste pagine, sarà possibile scrivere programmi Basic o in linguaggio macchina (attenzione a non realizzare pericolose sovrapposizioni!) e registrarli con una delle due forme sintattiche seguenti:

*Sys 15818 "Nome programma"*

*Sys 15818 "Nome", Inizio, Fine*

La prima forma è relativa a programmi Basic "standard", la seconda relativa ad una porzione di memoria compresa tra gli indirizzi "Inizio" e "Fine".

La particolarità della procedura risiede nel fatto che i programmi registrati con l'utility proposta possono esser caricati, in seguito, con un semplice Load, anche dopo aver spento e riacceso il computer.

In altre parole il programma pubblicato serve solo per la registrazione: per il caricamento non è necessaria la sua presenza nella memoria del computer.

**SCHEDA TECNICA**

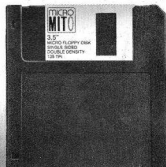
Software applicativo di utilità che velocizza le registrazioni ed i caricamenti di programmi.

Idoneo per computer C/16 e difficilmente adattabile ad altri computer Commodore

Richiede il registratore

Consigliato a coloro che non intendono rinunciare al registratore.

```
10 POKE 55,192:POKE 56,61:CLR
   :PRINTCHR$(147)
20 PRINT"SPEED TAPE SCRITTO  D
   A"
30 PRINT "SANTE MERCURIO. ATTE
   NDERE PREGO!"
40 FOR T=15818 TO 16356:READ A
   :POKE T,A:CK=CK+A:NEXTT
50 IF CK<>68003 THEN PRINT:PRI
   NT:PRINT"ERRORE NELLE LINEE
    DATA":END
60 PRINTCHR$(147)"SINTASSI DEL
    COMANDO:"
70 PRINT"SYS15818"+CHR$(34)+"N
   OMEFILE"+CHR$(34)+",INIZIO,
   FINE":PRINT
10000 DATA 162,4,181,42,149,154,2
   02,208
10010 DATA 249,32,121,4,162,0,201
   ,34
10020 DATA 208,18,160,0,230,59,17
   7,59
10030 DATA 201,34,240,8,157,42,63
   ,232
10040 DATA 224,16,208,238,224,16,
   176,9
10050 DATA 169,32,157,42,63,232,2
   4,144
10060 DATA 243,32,115,4,201,44,20
   8,26
10070 DATA 32,115,4,32,20,147,32,
   228
10080 DATA 157,132,155,133,156,32
   ,145,148
10090 DATA 32,20,147,32,228,157,1
   32,157
10100 DATA 133,158,165,156,72,165
   ,155,72
10110 DATA 165,158,72,165,157,72,
   169,192
10120 DATA 162,42,160,63,32,189,2
   55,169
10130 DATA 1,170,168,32,186,255,1
   69,122
10140 DATA 141,42,3,169,3,141,43,
   3
10150 DATA 162,42,134,178,133,179
   ,133,158
10160 DATA 162,45,134,157,70,154,
   32,164
10170 DATA 241,6,154,32,206,242,1
   69,16
10180 DATA 133,171,32,44,242,104,
   133,157
10190 DATA 104,133,158,104,133,15
   5,133,178
10200 DATA 104,133,156,133,179,17
   3,19,255
10210 DATA 9,2,141,19,255,160,0,1
   40
10220 DATA 252,7,173,6,255,41,239
   ,141
10230 DATA 6,255,202,208,253,136,
   208,250
10240 DATA 120,169,2,133,209,160,
   0,169
10250 DATA 2,32,252,62,162,7,136,
   192
10260 DATA 9,208,244,162,5,198,20
   9,208
10270 DATA 238,152,32,252,62,162,
   7,136
10280 DATA 208,247,162,5,152,32,2
   52,62
10290 DATA 165,178,32,252,62,165,
   179,32
10300 DATA 252,62,165,157,32,252,
   62,165
10310 DATA 158,32,252,62,132,215,
   162,7
10320 DATA 177,155,32,252,62,162,
   3,230
10330 DATA 155,208,4,202,202,230,
   156,165
10340 DATA 155,197,157,165,156,22
   9,158,144
10350 DATA 231,165,215,32,252,62,
   162,7
10360 DATA 136,208,246,206,252,7,
   88,173
10370 DATA 19,255,41,253,141,19,2
   55,76
10380 DATA 200,232,133,208,69,215
   ,133,215
10390 DATA 169,8,133,210,6,208,16
   5,1
10400 DATA 41,253,32,29,63,162,17
   ,9
```

# LA PERFEZIONE DIVENTA MITO

**QUAD-MITO** - 5 1/4" 96 TPI DS/QD
Floppy disk a quadrupla densità, disegnato per aumentare la capacità di registrazione sino a 780 kb per dischetto.
Velocità di registrazione 5800 BPI

**MEGA-MITO** - 5 1/4" 96 TPI HIGH DENSITY
Floppy ad alta densità, disegnato per drive da 1.2 MEG (AT e compatibili).
Velocità di registrazione 9650 BPI

**MICRO-MITO** - 3 1/2" 135 TPI DS/DD
Costruito per l'era dei disk drive da 3 1/2".
Velocità di registrazione 8100 BPI

le misure
della perfezione

**La *MICROFORUM MANUFACTURING INC.***
**è interessata all'ampliamento della propria rete distributiva.**
**Per qualsiasi contatto scrivere anche in italiano.**

944/A St. Claire Ave. West TORONTO, CANADA M6C 1C8
Tel. (416) 656-6406 - Tlx. 06-23303 MICROFORUM TOR
Telefax (416) 656-6368

```
10410 DATA 2,32,29,63,162,14,198,
      210
10420 DATA 208,234,96,202,208,253
      ,144,5
10430 DATA 162,11,202,208,253,133
      ,1,96
10440 DATA 83,80,69,69,68,32,84,4
      6
10450 DATA 49,53,56,49,56,32,32,3
      2
10460 DATA 0,169,16,36,1,208,252,
      36
10470 DATA 1,240,252,173,9,255,72
      ,169
10480 DATA 7,141,4,255,142,5,255,
      169
10490 DATA 64,141,9,255,104,10,10
      ,96
10500 DATA 169,8,133,210,32,72,3,
      38
10510 DATA 208,238,25,255,198,210
      ,208,244
10520 DATA 165,208,96,32,206,242,
      169,198
10530 DATA 133,1,32,100,227,120,1
      69,7
10540 DATA 141,4,255,162,1,32,72,
      3
10550 DATA 38,208,165,208,201,2,2
      08,245
10560 DATA 160,9,32,103,3,201,2,2
      40
10570 DATA 249,196,208,208,232,32
      ,103,3
10580 DATA 136,208,246,32,103,3,1
      53,155
10590 DATA 0,200,192,4,208,245,16
      0,0
10600 DATA 132,215,32,103,3,145,1
      55,69
10610 DATA 215,133,215,230,155,20
      8,2,230
10620 DATA 156,165,155,197,157,16
      5,156,229
10630 DATA 158,144,231,32,103,3,7
      2,32
10640 DATA 200,232,104,197,215,24
      0,3,76
10650 DATA 43,168,166,157,164,158
      ,134,45
10660 DATA 132,46,32,24,136,32,15
      4,138
10670 DATA 76,3,135
10680 END
```

# La parola all'Amiga

*Per molti la sintesi vocale dell'Amiga non è altro che un simpatico gadget. In realtà può essere utilizzata anche per altri scopi: vediamo come*

di Michele Maggi

La caratteristica dell'Amiga, che più di ogni altra colpisce l'attenzione del nuovo utente, è senz'altro la sintesi vocale, grazie soprattutto alla facilità d'uso.

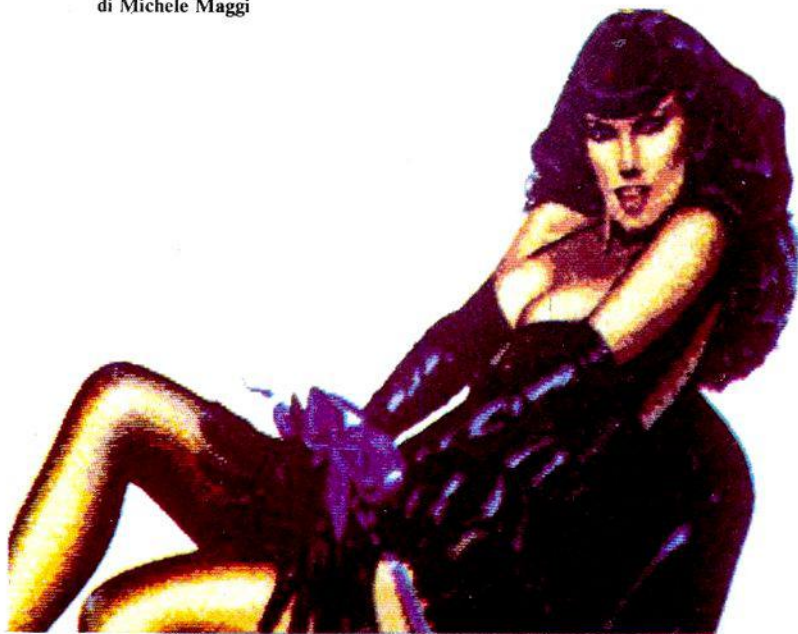
Per i "sessantaquattristi" abituati al S.A.M. o alla Voce (versioni I, II e III), il salto di qualità è notevole, se non addirittura sorprendente.

La fedeltà di riproduzione della voce umana è quasi perfetta e l'unico neo è rappresentato dal fatto che la pronuncia, invece che essere in un italiano degno di una graziosa annunciatrice, si avvicina (ahimè) alla parlata (per alcuni simpatica) di Dan Peterson alle prese con un commento sul Basket NBA (fortunatamente senza "Uhhhh U'!" finale della pubblicità).

Chiunque debba, per motivi di lavoro, studio o diletto, cimentarsi con testi inglesi o americani dei quali ignora la pronuncia, potrà ottenere, grazie appunto alla sintesi vocale, un aiuto per meglio imparare a pronunciare i vocaboli in questione.

E' presto detto che il programma "Demo Speech" incluso nel disco Extras non assolve a questo scopo anche se riesce comunque a far intravedere le notevoli potenzialità del sintetizzatore facilmente controllabile grazie alla potente istruzione SAY.

## Due esperimenti

Dopo aver caricato AmigaBASIC proviamo a digitare in modo diretto (=nella finestra Basic) il seguente comando:

```
SAY TRANSLATE$ ("I love my new Commodore Amiga")
```

Verrà richiesto di inserire il disco Workbench da cui il sistema operativo (S.O.) caricherà un programma chiamato "Narrator Device" necessario per la pronuncia del testo.

Solo a questo punto (finalmente!) udremo la voce della nostra Amiga.

Tutto chiaro per l'istruzione SAY, ma a che cosa serve, dunque, la funzione TRANSLATE$ ?

Proviamo ora a digitare il seguente gruppo di comandi:

```
A$="COMMODORE AMIGA"
PRINT TRANSLATE$ (A$)
SAY TRANSLATE$ (A$)
```

Ciò che vedremo stampato sarà la seguente stringa di caratteri:

KAA3MAXDOHR AHMIY3GAH

che è ciò che realmente l'Amiga pronuncia tramite SAY.

In sostanza TRANSLATE$ funge da "convertitore" del testo in "fonemi" pronunciabili dall'Amiga; tutto ciò si risolve in una notevole facilitazione per ciò che riguarda la programmazione.

Ma le "features" del sintetizzatore non finiscono certo qui: tramite la gestione di nove parametri è infatti possibile modificare l'inflessione, la velocità e altre cose ancora.

## I nove parametri

Nella tabella che segue vengono evidenziati i nove parametri con i relativi valori minimi, massimi e di default necessari per modificare le caratteristiche della pronuncia.

| Argomento | Min. | Def. | Max. |
|---|---|---|---|
| Pitch | 65 | 110 | 320 |
| Inflection | 0 | 0 | 1 |
| Rate | 40 | 150 | 400 |
| Voice | 0 | 0 | 1 |
| Tuning | 5000 | 22200 | 28000 |
| Volume | 0 | 64 | 64 |
| Channel | 0 | 10 | 11 |
| Mode | 0 | 0 | 1 |
| Control | 0 | 0 | 2 |

Vediamo ora come è possibile modificare la pronuncia tramite la loro "manipolazione". Il primo parametro, PITCH, modifica l'altezza del timbro della voce ed il valore di default, fissato a 110, simula l'altezza media di una voce maschile.

Il parametro INFLECTION rende la voce naturalmente espressiva (0) o inespressiva (1) come se il testo fosse pronunciato da un robot.

Il terzo parametro, RATE, rappresenta la velocità con cui il sintetizzatore pronuncia le parole, mentre il quarto, VOICE, rende la voce maschile (0) oppure femminile (1)

TUNING controlla la frequenza di campionamento in hertz e, indirettamente, anche l'altezza della voce, VOLUME ne controlla il volume d'uscita, mentre il parametro CHANNEL indirizza la voce ad un canale piuttosto che ad un altro oppure ad entrambi (come nel caso del valore di default).

Gli ultimi due parametri, MODE e CONTROL, servono per la sincronizzazione delle voci, argomento, questo, che tratteremo per motivi di spazio in un prossimo articolo; per il momento andranno utilizzati tramite i valori di default.

Vediamo ora in che modo è possibile "passare" al sintetizzatore i parametri.

E' opportuno costruire un vettore di numeri interi (per esempio: Voice%) con i nove valori relativi ai parametri; in AmigaBASIC può essere utilizzata la seguente procedura:

```
DIM Voice% (8)
FOR I=0 TO 8
READ Voice%(I)
NEXT
DATA 160, 0, 180, 0, 22000, 64, 10,
0, 0
```

Se modifichiamo la sintassi di SAY con...

```
SAY TRANSLATE$ ("A new voice"),Voice%
```

...otterremo una pronuncia modificata rispetto a quella standard.

E' senz'altro utile effettuare diverse prove per comprendere a fondo in che modo i vari parametri, e soprattutto le loro combinazioni, possono "creare" una voce.

## Amiga Reciter V 1.0

Un esempio di applicazione pratica di quanto detto è il programma presentato in queste pagine, in cui Amiga Reciter legge e pronuncia un testo scritto con il programma "Notepad".

Chi non avesse dimestichezza con la sopracitata utility può dare un'occhiata al riquadro.

Lo spirito del programma è quello di permettere di udire, con una pronuncia il più possibile simile a quella inglese originale, la lettura di un brano scritto, appunto, con Notepad.

I passi da seguire sono i seguenti:

- Caricare Notepad
- Scrivere il testo inglese
- Salvare il testo
- Uscire da Notepad
- Caricare Amiga Reciter

## Funzionamento del programma

La prima cosa che salta all'occhio è la mancanza dei numeri di linea che (per fortuna!) sono opzionali in AmigaBASIC.

Ciò significa che i GOTO e i GOSUB potranno essere indirizzati verso label piuttosto che verso numeri di linea veri e propri.

Le label (o etichette) non sono altro che gli "indirizzi" di determinate sot-

**Come usare Notepad**

Questa breve nota è dedicata a chi intenda usare "Notepad" per lo scopo suindicato; per una trattazione completa ed esauriente si rimanda ad un prossimo articolo espressamente dedicato all'utility in questione.

1/ Clickare sull'icona di Notepad
2/ Attendere il caricamento e la lettura dei font
3/ Stabilire la grandezza della finestra
4/ Digitare il testo (fino a un massimo di 10 pagine). Ricordarsi di scrivere in inglese per usare il programma di queste pagine (pena una ridicola emissione di parole senza senso)
5/ Premere il bottone destro del mouse per selezionare l'opzione "SAVE AS" dal menu "PROJECT"
6/ Digitare il nome del documento (e premere Return)
7/ Uscire da Notepad

toprocedure all'interno di un programma; l'uso di label al posto di numeri di linea rende molto più leggibile (e soprattutto comprensibile) il testo di un programma Basic, perchè il programmatore tenderà ad assegnare nomi "logici" alle label; ad esempio una procedura di inizializzazione si chiamerà quasi sicuramente INIZIALIZZA o qualcosa di simile.

Un GOSUB INITIALIZE, insomma, rende molto meglio l'idea rispetto ad un GOSUB 15500, a tutto vantaggio della leggibilità del programma.

Va notato che le label non possono essere uguali a parole chiave del Basic e che devono essere SEMPRE seguite dal carattere dei due punti (:).

Ma torniamo al programma...

La prima linea, contenendo l'istruzione SAY, obbligherà il computer a caricare il "Narrator device" da Workbench; se avete due (o più) disk drive, in uno dei quali lasciare permanentemente Workbench, non farete la fatica di sostituire il disco Extras nel drive interno.

L'Amiga leggerà in seguito, il file, carattere dopo carattere, concatenandoli in una stringa finchè non verrà incontrato uno dei quattro separatori; quindi pronuncerà la stringa.

I separatori stabiliti nel programma pubblicato sono il punto (.), il punto interrogativo (?), quello esclamativo (!) e i due punti (:); i loro codici ASCII sono contenuti nella prima linea DATA.

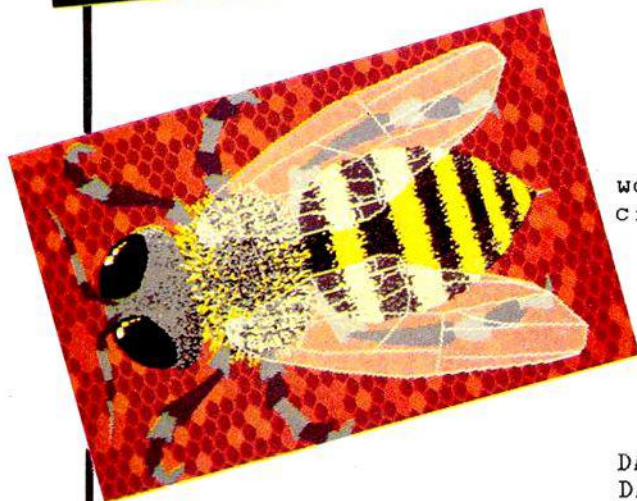
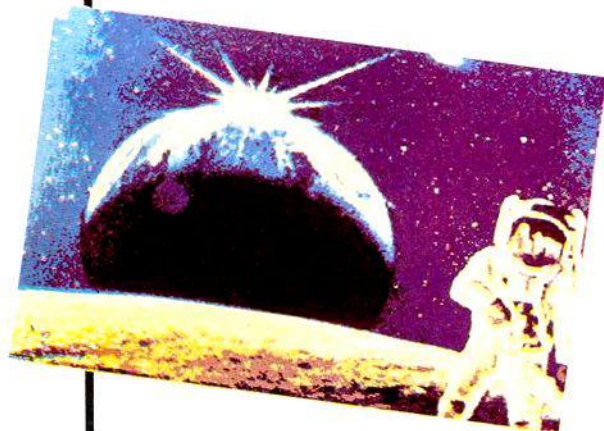
Vengono letti tramite il primo ciclo FOR... NEXT e salvati nella matrice "Separatore (x)". In questo caso i separatori sono solo quattro ma è possibile aggiungerne altri, in funzione delle proprie esigenze.

Successivamente (vedi seconda linea DATA) viene creato l'array Voice% per creare un certo timbro di voce (inutile dire che tutti i parametri sono facilmente modificabili), quindi si passa all'input per il nome del file.

Il nome deve essere eventualmente preceduto dal numero del drive e/o dal nome della directory madre.

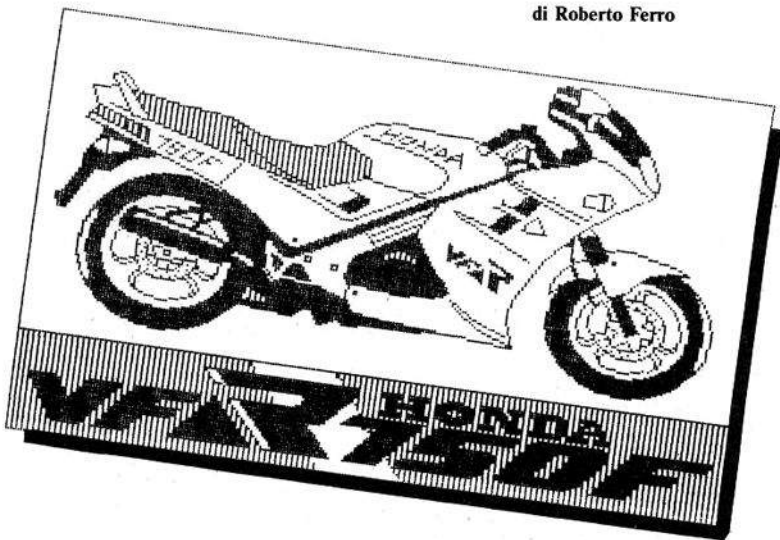
Il corpo centrale del programma riguarda la lettura dei caratteri da disco e la loro pronuncia.

Ogni 50 caratteri il controllo passa alla routine Wordwrap che manda a capo per evitare "sconfinamenti".
La routine "Finisce" provoca un Beep e stampa in reverse il messaggio di fine file.

Il listato si presta a notevoli sofisticazioni e anche il lettore meno esperto potrà comprenderne il funzionamento.

```
'---------------------
' Amiga Reciter V 1.0
'        1-7-87
'   by Michele Maggi
'---------------------

SAY TRANSLATE$ ("welcome!")
FOR x=1 TO 4
 READ separatore (x)
NEXT
DIM voice%(8)
FOR y=0 TO 8
 READ voice%(y)
NEXT
 INPUT "nome completo del file";F$
OPEN F$ FOR INPUT AS 1
WHILE NOT EOF(1)
 a$=INPUT$ (1,#1)
 IF a$= CHR$(10) THEN ct=0
  IF a$=CHR$(12) THEN a$=""
   PRINT a$;:s$=s$+a$
 FOR x=1 TO 4
  IF a$ = CHR$(separatore (x)) THEN
   SAY TRANSLATE$(s$),voice%
  s$=""
  END IF
 NEXT
 ct=ct+1
 IF ct=50 THEN GOSUB wordwrap
WEND
 finisce:
     BEEP
     COLOR 0,1
     PRINT
     PRINT  "--- fine del file ---"
     COLOR 1,0
     END

wordwrap:
ciclo:   a$=INPUT$ (1,#1)
         PRINT a$;:s$=s$+a$
         IF a$= CHR$(32) THEN
          PRINT CHR$ (13);
           ct=0
            RETURN
           END IF
          GOTO ciclo

DATA 58,46,33,63
DATA 80,0,140,0,24000,64,10,0,0
```

# Artisti si diventa

*Poco più di quattro schermate di listato per sfruttare a fondo le potenti istruzioni grafiche del Basic 3.5 e 7.0*

di Roberto Ferro

**SCHEDA TECNICA**

Software idoneo per applicazioni grafiche.

Idoneo per computer: Plus/4 e C/128; funziona sul C/16 inespanso a patto di attivare l'espansione Ram video.

Adattabile, pur se con qualche difficoltà, ad altri computer Commodore.

Richiede il joystick oppure il mouse.

E' previsto l'uso del disk drive.

Consigliato a tutti gli appassionati di grafica.

Anche i programmi pubblicati in queste pagine (oltre al alcuni file applicativi) sono contenuti nel disco "Directory" di questo mese.

Il programma, chiamato "Artista", funzionante sul Plus 4 come sul C/128, è forse il primo a sfruttare pienamente le risorse offerte da un'eventuale espansione (software) di memoria suggerita per il C/16.

Infatti il programma, che gira tranquillamente su Plus 4 e C/128, non può funzionare su C/16, se non con l'espansione di cui abbiamo avuto modo di occuparci su Commodore Computer Club: dopo averlo digitato, va salvato senza essere lanciato (pena un "Out of memory error"). Per utilizzarlo bisogna attivare dapprima l'espansione, lanciarla, quindi caricare "Artista" e, finalmente, farlo girare.

Artista è finalizzato alla produzione di disegni, anche di precisione, da salvare su nastro (o disco) per essere richiamati anche indipendentemente dal programma.

La motocicletta che compare in queste pagine è un esempio delle possibilità offerte dal programma.

## Funzionamento

L'uso è piuttosto semplice e permette di selezionare tutte le opzioni senza mai uscire dalla pagina grafica.

Il programma prevede l'uso del joystick (o del mouse) in porta 1 per muovere il cursore e attivare le opzioni, e dei tasti cursore sinistro e destro per selezionarle (per il C/128 usare i tasti cursore posizionati in alto).

Con il pulsante di fire si fissano i punti di inizio e di fine di ogni figura. Il messaggio costantemente presente in alto a sinistra ricorda con quale opzione stiamo lavorando.

Le scelte possibili sono:

• ***Elastico.*** Permette di disegnare una linea che, partendo da un punto prefissato, può essere posizionata a piacere prima di premere nuovamente fire e di fissarla definitivamente sullo schermo. La linea può servire anche per cancellare aree di disegno: dopo averla fatta passare dove intendiamo cancellare la potremo richiudere sul punto di partenza.

• ***Riempimento.*** Come dice il nome stesso è utile per riempire una figura dai contorni ben definiti; fate attenzione a non lasciare aperture, pena la colorazione totale e irreversibile del video!

Lavorando con questa opzione sa-

rà udibile un rumore che ha lo scopo di allarmare l'utilizzatore, impedendo manovre pericolose.

• ***Mano libera.*** Premendo il tasto fire, il cursore lascerà, dietro di sè, una scia che potremo interrompere con una successiva pressione del tasto in modo da spostarci altrove senza lasciar traccia.

• ***Cerchi.*** Per tracciare un cerchio dovremo portarci nel centro, premere fire e allontanarci tanto in direzione destra o sinistra (asse X) quanto in quella alto o basso (asse Y), determinando, in tal modo, i raggi del cerchio o dell'ellisse. A questo punto, dopo aver visualizzato il cerchio fissando il secondo dei due punti, possiamo modificare le dimensioni della figura tracciata muovendoci ulteriormente. Se volessimo cancellare il cerchio appena tracciato sarà sufficiente, nel momento in cui concludiamo, tenere il tasto di fire premuto per qualche secondo, finchè la figura non viene cancellata.

• ***Rettangoli.*** Il funzionamento di questa opzione è semplice: si fissano i vertici del rettangolo che, una volta tracciato, potremo modificare muovendo il joystick. Per terminare occorre premere il fire.

• ***Spruzzo (aerografo).*** Serve per "spruzzare" casualmente alcuni puntini nei pressi del cursore. Grazie a questa opzione è possibile creare effetti di chiaro-scuro: più a lungo si terrà premuto il tasto fire, maggiore sarà la densità dei puntini, proprio come se si usasse un aerografo!

• ***Taglia.*** Ha la funzione di memorizzare (piccole) aree del disegno: dovremo fissare i punti in alto,a sinistra e in basso a destra della zona da ritagliare (o meglio, copiare). Successivamente potremo spostarci sullo schermo e "provare" a sistemare la nostra figura premendo il tasto fire; rilasciandolo, la figura si cancellerà lasciando inalterato lo sfondo precedente. Se l'area che tenteremo di salvare avrà una superficie maggiore di 900 pixel (1600 per C/128 e Plus 4), uno sgradevole rumore avvertirà che non è possibile copiare una zona tanto vasta.

Se volessimo ritagliare un'altra area sarà sufficiente cancellare dalla memoria la zona precedentemente prelevata tenendo il fire premuto qualche secondo, come per provarla.

E' INDISPENSABILE che una zona ritagliata sia incollata oppure cancellata PRIMA di attivare altre opzioni; in caso contrario si rischia di non poter più fruire di questa opzione!

• ***Incolla.*** Incolla la figura ritagliata sovrapponendola allo sfondo. E' possibile riprodurre infinite volte la figura memorizzata mediante "Taglia".

• ***Incolla OR.*** Incolla la figura ritagliata praticando un OR tra questa e lo sfondo; serve per sovrapporre l'immagine memorizzata mediante "Taglia" alla zona di scehrmo di eguali dimensioni.

• ***Incolla RVS.*** Riproduce in reverse (nero invece di bianco e viceversa) la figura memorizzata.

• ***Coordinate.*** Per lavori di precisione è necessario sapere di quanti pixel ci si muove. Questa opzione, una volta attivata premendo il tasto fire, permette di conoscere, in qualunque modalità, di quanti pixel ci siamo spostati dal punto di partenza. Per disattivarla è sufficiente premere nuovamente fire.

• ***Salva.*** E' utile per registrare il disegno su nastro. Verrà chiesto il nome del disegno e, subito dopo, comparirà il messaggio tipico del Monitor in Linguaggio Macchina.

A questo punto si dovrà, nell'ordine, premere il tasto F1; compiere il salvataggio; premere il tasto F2.

• ***Carica.*** Serve a richiamare un disegno da nastro. Le operazioni da compiere sono le stesse di "Salva", tranne la richiesta del nome; è quindi necessario posizionare il nastro il modo opportuno.

• ***Cancella.*** Con questa opzione possiamo ricominciare daccapo: non sarà più possibile recuperare il disegno cancellato. Per evitare le conseguenze di spiacevoli distrazioni viene reso udibile lo stesso rumore dell'opzione "Riempimento". Inoltre, per cancellare il disegno, bisognerà tener premuto il fire per qualche istante.

## Qualche altra informazione

La struttura del programma è piuttosto semplice e si basa sul comando ON...GOTO che lo rende molto veloce e comprensibile.

I punti nevralgici sono pochi se si escludono le operazioni di SAVE e LOAD che azzerano il buffer di tastiera per evitare che i caratteri immessi con il joystick si "scarichino" su video; si provvede, inoltre, alla ridefinizione di due tasti funzione (F1 e F2).

Chi volesse utilizzare il drive come memoria di massa, deve modificare le righe 330 e 350 come segue:

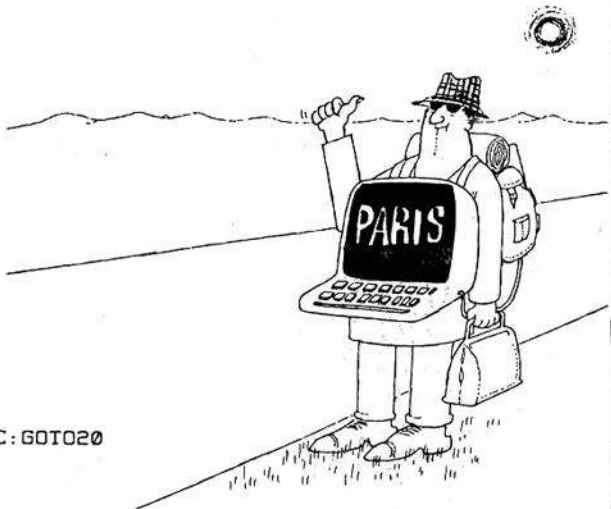
*330 KEY1,"S" + CHR$(34) + NM$ + CHR$(34) + ",8,2140,3F41" + CHR$(13)*
*350 GRAPHIC0,1: INPUT"NOME";NM$: KEY1, "L" + CHR$(34) + NM$ + CHR$(34) + ",8" + CHR$(13): GOTO 335*

Ogni pagina grafica occuperà **40** blocchi del disco.

Nota anti-panico per i possessori di C/16: è possibile (ma improbabile) che in qualche particolare passaggio, soprattutto con "Taglia" e "Incolla", si verifichi una condizione di Out of memory error. In questo malaugurato caso sarà sufficiente un **RUN 10** per riprendere tranquillamente il lavoro.

## Per il C/128

Le modifiche da apportare per il C/128 sono relative ad alcune POKE. Ad esempio la locazione che scandisce la tastiera sul C/16 è la 198 mentre è la 212 sul C/128. Altre piccole differenze risiedono nella gestione dei colori e nell'estensione della superficie di "Taglia Incolla" (maggiore sul C/128 e Plus 4 grazie alla maggior quantità di memoria). Un'altra modifica può essere apportata al comando SOUND la cui frequenza va raddoppiata rispetto a quella del C/16.

```
5 COLOR4,2:COLOR0,2:COLOR1,1:GRAPHIC1,1
10 GRAPHIC1,0:A=160:B=100:F=1:GOTO80
20 IFF=3ORF=14THENVOL4:SOUND3,800,5
30 IFPEEK(198)=51ANDK=0THEN75
35 IFPEEK(198)=48ANDK=0THEN65
40 J=JOY(1):ONJGOTO100,105,110,115,120,125,130,135
50 IFJ>127ANDPEEK(198)=22THEN55:ELSE20
55 IF(F=4ORF=5)ANDU=1THEN60:ELSEP=A:N=B
60 ONFGOTO175,210,190,220,295,195,370,410,405,415,270,345,320,360
65 F=F-1:IFF=0THENF=14
70 GOTO80
75 F=F+1:IFF=15THENF=1
80 FORI=1TOF:READA$:NEXT
85 A$=A$+"       ":CHAR1,1,0,A$
90 VOL8:SOUND1,810,5:RESTORE:GOTO20
100 X=-1:GOTO140
105 X=-1:Y=1:GOTO140
110 Y=1:GOTO140
115 Y=1:X=1:GOTO140
120 X=1:GOTO140
125 X=1:Y=-1:GOTO140
130 Y=-1:GOTO140
135 Y=-1:X=-1
140 DRAWC,A,B:A=A+Y:B=B+X
145 LOCATEA,B:C=RDOT(2)
150 IFC=1THEND=0:ELSED=1
155 ONKGOSUB240,250,260,315
160 DRAWD,A,B:IFT=1THENGOSUB285
170 X=0:Y=0:GOTO40
175 GOSUB420:IFK=0THENK=2:E=A:G=B:L=C:GOTO20
180 IFE=AANDG=BTHENC=L:ELSEC=1
185 K=0:GOTO20
190 DRAW0,A,B:PAINT1,A,B:C=1:GOTO20
195 DOWHILEJOY(1)>127
200 S=(RND(1)*8)-4:V=(RND(1)*8)-4:DRAW1,A+S,B+V:LOOP:GOTO20
205 LOOP:GOTO20
210 GOSUB420:IFK=3THENK=0:C=1:ELSEK=3
215 GOTO20
220 IFU=0THENU=1:L=A:M=B:SOUND1,700,7:GOSUB420:GOTO20
225 IFU=1THENE=ABS(A-L):G=ABS(B-M):CIRCLE1,L,M,E,G:K=1:U=2:GOSUB420:GOTO20
230 IFU=2THENK=0:U=0:SOUND1,700,7:DOWHILEJOY(1)>127:O=O+1:IFO>100THENCIRCLE0,L,M
,E,G
235 LOOP:O=0:GOTO20
240 CIRCLE0,L,M,ABS(A-L-Y),ABS(B-M-X):LOCATEA,B:C=RDOT(2)
245 E=ABS(A-L):G=ABS(B-M):CIRCLE1,L,M,E,G:RETURN
250 DRAW0,A-Y,B-XTOE,G:DRAW1,A,BTOE,G:RETURN
260 DRAW1,A-Y,B-X:RETURN
270 GOSUB420:IFT=1THENT=0:CHAR1,20,0,"                              ":GOTO280
275 T=1:CHAR1,20,0,"X = 0        Y = 0  "
280 SOUND1,400,9:GOTO20
285 Q=(N-B):R=(A-P):B$=STR$(R)+"  ":C$=STR$(Q)+"  ":CHAR1,23,0,B$:CHAR1,35,0,C$
290 RETURN
295 GOSUB420:IFU=0THENU=1:L=A:M=B:C=1:SOUND1,700,7:GOTO20
300 IFU=1THENBOX1,L,M,A,B:U=2:K=4:GOTO20
305 IFU=2THENC=1:K=0:U=0:SOUND1,700,7:GOTO20
315 BOX0,L,M,A-Y,B-X:BOX1,L,M,A,B:RETURN
320 FORI=1319TO1328:POKEI,0:NEXT
325 GRAPHIC0,1:INPUT"NOME";NM$
330 KEY1,"S"+CHR$(34)+NM$+CHR$(34)+",1,2140,3F41"+CHR$(13)
335 KEY2,"X"+CHR$(13)+"RUN10"+CHR$(13):MONITOR
345 FORI=1319TO1328:POKEI,0:NEXT
```

```
350 GRAPHIC0,1:KEY1,"L"+CHR$(13):GOTO335
360 DOWHILEJOY(1)>127:O=O+1:IFO=200THENRUN:ELSELOOP:GOTO20
370 IFU=0THENU=1:L=A:M=B:SOUND1,700,7:GOSUB420:GOTO20
375 IFU=1THENU=2:E=A:G=B:ELSE390
380 IF(E-L)*(G-M)>900THENSOUND3,99,30:U=0:GOTO400
385 SOUND1,700,7:SSHAPES$,L,M,E-1,G:GOSUB420:GOTO20
390 IFU=2THENSSHAPEG$,A,B,A+(E-L),B+(G-M)
395 GSHAPES$,A,B,2:DOWHILEJOY(1)>127:O=O+1:IFO>200THENU=0:EXIT:ELSELOOP
400 O=0:GSHAPEG$,A,B:G$="":GOSUB420:GOTO20
405 GSHAPES$,A+1,B,2:U=0:GOTO20
410 GSHAPES$,A+1,B:U=0:GOTO20
415 GSHAPES$,A+1,B,1:U=0:GOTO20
420 DOWHILEJOY(1)>127:LOOP:RETURN
1000 DATA1-ELASTICO,2-MANO LIBERA
1005 DATA3-RIEMPIMENTO,4-CERCHI
1010 DATA5-RETTANGOLI,6-SPRUZZO
1015 DATA7-TAGLIA,8-INCOLLA
1020 DATA9-INCOLLA OR,10-INCOLLA RVS
1025 DATA11-COORDINATE,12-CARICA
1030 DATA13-SALVA,14-CANCELLA

READY.
```

```
0 REM MODIFICHE DA APPORTARE AL
1 REM PROGRAMMA "ARTISTA" (VERSIONE C/128)
2 :
5 COLOR4,2:COLOR0,2:COLOR1,1:COLOR5,1:GRAPHIC1,1
30 IFPEEK(212)=86ANDK=0THEN75
35 IFPEEK(212)=85ANDK=0THEN65
50 IFJ>127THEN55:ELSE20
195 DOWHILEJOY(1)>127:SOUND1,65000,3,,,3
320 FORI=842TO851:POKEI,0:NEXT:POKE209,0
345 FORI=842TO851:POKEI,0:NEXT:POKE209,0
380 IF(E-L)*(G-M)>1600THENSOUND3,198,30:U=0:GOTO400

READY.
```

# Tutte le lingue del nostro computer

*Una carrellata sui vari sistemi per mettersi in contatto con un calcolatore*

di Paolo Agostini

Chi compra un impianto stereofonico, oppure un videoregistratore, deve imparare solo ad usare i tasti principali per far funzionare l'apparecchio.

Nel caso di un computer la cosa è invece diversa. Per prima cosa il novello utente dovrà rendersi conto a cosa servono i numerosi tasti e, dopo esser riuscito con fatica e difficoltà (vista anche la qualità dei manuali forniti dalla casa costruttrice!) si trova soltanto all'inizio dell'opra sua. A quel punto deve apprendere una nuova lingua, il BASIC, per programmare il nuovo "elettrodomestico".

Dopo un primo momento di euforia, quando si crede che col Basic non ci sia nulla di precluso, ci si rende conto degli svantaggi di questo linguaggio di programmazione: i programmi in Basic sono lenti, non consentono il controllo assoluto della macchina, la "chiamata" di subroutines mediante numeri di linea, anzichè per nome, e la mancanza di comandi per la programmazione strutturata, portano inevitabilmente a una mancanza di leggibilità del listato (gli americani parlano, nel caso di listati particolarmente "impenetrabili" di Spaghetti-Code!). Inoltre

non esiste un solo Basic, ma molti, ognuno con caratteristiche proprie, il che rende spesso impossibile trascrivere il programma su una macchina diversa da quella per la quale è stato progettato; manca, cioè, la dote di "portabilità".

Esistono, infatti, moltissimi cosiddetti "dialetti" Basic, vale a dire versioni che si discostano dallo standard, che rendono impossibile, per esempio, il poter utilizzare senza modifiche sul C-64 un programma scritto col Basic Apple.

Di solito gli hobbisti, raggiunta una certa preparazione, decidono di abbandonare il Basic per rivolgersi al "linguaggio macchina" che sembra possedere qualcosa di magico per i novelli alchimisti.

Ogni computer ha un suo "linguaggio macchina" (L.M.), ovvero il linguaggio del microprocessore impiegato. Il LM altro non è se non una serie interminabile di "0" e di "1" (o meglio di "0" inteso come "non c'è corrente" e "1" come "sta passando corrente"), che è difficilmente comprensibile per gli esseri umani. Per questa ragione è stato inventato il linguaggio ASSEMBLY che rappresenta un mezzo ideale per la rappresentazione dei comandi in linguaggio macchina. Un apposito programma, detto ASSEMBLER trasforma i codici Mnemonici (comandi che il programmatore può ricordare con facilità) del linguaggio Assembly in una sequenza di bit "0" e "1" che rappresenta il linguaggio macchina vero e proprio. Linguaggio Assembly e linguaggio macchina sono però così strettamente collegati che oggigiorno vengono considerati sinonimi.

Ma esistono delle alternative alle fatiche di una faticosa programmazione in LM bit per bit: ci si dimentica spesso dell'esistenza di linguaggi di programmazione, diversi dal Basic, che consentono di raggiungere egualmente lo scopo.

Il presente articolo si propone di mostrare quali siano i linguaggi principali di programmazione (e quelli disponibili per il Commodore 64 e 128), e l'impiego per il quale tali linguaggi sono particolarmente adatti.

## Un po' di storia

In principio vi era il linguaggio macchina e soltanto all'inizio degli anni '50 si cominciò a parlare di "linguaggi ad alto livello". Il linguaggio che la macchina "capisce" è sensibilmente differente da un linguaggio ad "alto livello", il quale è, o quanto meno dovrebbe essere, indipendente dal tipo di processore usato e dovrebbe poter girare senza problemi su macchine differenti.

Il linguaggio ad "alto livello" dovrebbe essere di più facile apprendimento del linguaggio macchina e di più facile comprensione e lettura per l'utente umano; inoltre il linguaggio ad alto livello dovrebbe essere in grado di ridurre sensibilmente la possibilità di errori che impediscano il funzionamento della macchina. Purtroppo non esiste alcun microprocessore in grado di accettare, ed eseguire, comandi dati ad un livello che è semplicemente troppo "alto" per lui. Il problema quindi viene risolto mediante l'utilizzo di un INTERPRETE o di un COMPILATORE che si incaricano di rendere comprensibili al processore i comandi stessi, traducendoli in "semplici" istruzioni in linguaggio macchina. Il processo di interpretazione avviene secondo regole ben precise e codificate da ALGORITMI.

La differenza tra compilatori e interpreti è piccola, ma sostanziale. Un compilatore è una utility che trasforma un programma scritto in un linguaggio ad "alto" livello in un programma in linguaggio macchina. Il compilatore viene usato una sola volta, al momento in cui la versione definitiva del programma scritto in un linguaggio ad "alto" livello viene tradotta in linguaggio macchina. Da quel momento in poi il programma "gira" senza più aver bisogno del compilatore. Pensiamo invece al C-64: esso possiede un interprete Basic che "simula" un super-processore col Basic come linguaggio macchina. Questa simulazione è pressochè perfetta: la macchina accetta input da tastiera e reagisce con prontezza alle nostre istruzioni. Ciò ha però un alto costo in termini di tempo: l'interprete Basic, infatti, deve prendere un carattere alla volta, "capire" di che cosa si tratta e, se si tratta di un comando, porlo in esecuzione. Se lo stesso comando deve essere eseguito dieci, cento o diecimila volte, l'interprete sarà costretto a "re-interpretarlo" dieci, cento o diecimila volte. Questa è la ragione per cui il Basic è lento.

L'inglese Charles Babbage può essere considerato il padre spirituale dei calcolatori, dal momento che iniziò a costruire meccanismi digitali nel lontano 1833. Le sue macchine erano fatte di ingranaggi, manovelle e leve, ma l'elettromeccanica, prima, e l'elettronica, poi, tramutarono quei primitivi dispositivi nei moderni calcolatori. Il primo calcolatore a relais fornito di programma venne costruito nel 1941 in Germania da Konrad Zuse, ed era chiamato Zuse Z3. Il primo calcolatore programmabile americano, anch'esso a relais, venne costruito nel 1944 dal matematico Howard H. Aiken per conto della IBM e aveva il nome di Mark I: era lungo 16 metri e pesava 35 tonnellate. Quando nel 1945 le tecniche elettroniche conobbero finalmente uno sviluppo sufficente, gli arcaici e lenti relais vennero soppiantati totalmente da tubi elettronici nel famoso ENIAC. I computer a valvole della prima generazione raggiunsero il loro apogeo col SSEC (Selective Sequence Electronic Calculator) verso la fine degli anni '40.

## Il Fortran

In quel periodo il numero di computer iniziò ad aumentare e gli studi sulla programmazione dei calcolatori portarono i primi frutti. Nel 1955 apparve un libro sulla programmazione nel più anziano dei linguaggi ad alto livello, il FORTRAN. Due anni più tardi questo linguaggio veniva implementato sul calcolatore IBM 704. Da quel momento il Fortran conobbe una notevole diffusione tra i programmatori che sino ad allora erano stati costretti a programmare in linguaggio macchina, con notevoli costi per i centri di calcolo. Il nome del Fortran è un acronimo di FORmula TRANslating System (=traduttore di formule), e proprio questo fu il punto di forza del linguaggio: i

più difficili problemi di calcolo possono essere risolti con facilità, tanto che il Fortran è ancora oggi, nonostante la sua età, uno dei linguaggi più importanti nei campi dell'ingegneria e della matematica.

Oltre ai numeri REALI (in virgola mobile), INTEGER (numeri interi) e BOOLEANI, il Fortran è in grado di fare calcoli in DOUBLE PRECISION per una maggiore esattezza dei risultati. Col passare degli anni apparvero versioni via via migliorate del linguaggio, che nelle ultime versioni (Fortran V o Fortran 77) offre comandi che ne permettono la strutturazione (IF THEN ELSE ENDIF) e che hanno contribuito a mitigare la sua "arcaicità". Il Fortran "gira" sotto CP/M; non ci risulta, però, che siano disponibili versioni per il C-128.

## L'Algol

Con l'ALGOL (ALGOrythmic Language) si era tentato di risolvere i problemi posti dall'elaborazione matematica, ma questo linguaggio non conobbe mai una diffusione notevole, benchè proprio da questo linguaggio derivi un notevole numero di altri linguaggi ad alto livello, non ultimo l'ADA di cui parleremo più avanti. L'Algol 60 offrì i primi concetti validi della programmazione strutturata.

## Il Cobol

Nel 1959 il Ministero della Difesa degli Stati Uniti commissionò, ad un comitato composto da qualche centinaio di "esperti", un linguaggio che avrebbe dovuto servire per usi commerciali: venne chiamato COBOL, acronimo di COmmon Business Oriented Language.

Uno degli scopi che ci si proponeva, con la creazione di questo linguaggio, era quello di far sì che anche personale non specializzato fosse in grado di leggere programmi. Da tali premesse nacque il linguaggio di programmazione più brutto e assurdo che esista: dotato di una verbosità incredibile e della necessità di inverosimili quantità di memoria, questo linguaggio è stato sin quasi ai giorni nostri il più diffuso nel mondo per utilizzazioni commerciali e soltanto in questi ultimi tempi inizia a perdere terreno in favore di linguaggi più agili, più efficienti e, perchè no, più aggraziati. I programmi in Cobol simulano la lingua inglese. Ad esempio, la riga Basic...:

*IF X>0 THEN A=B/X*

...viene tradotta in Cobol:

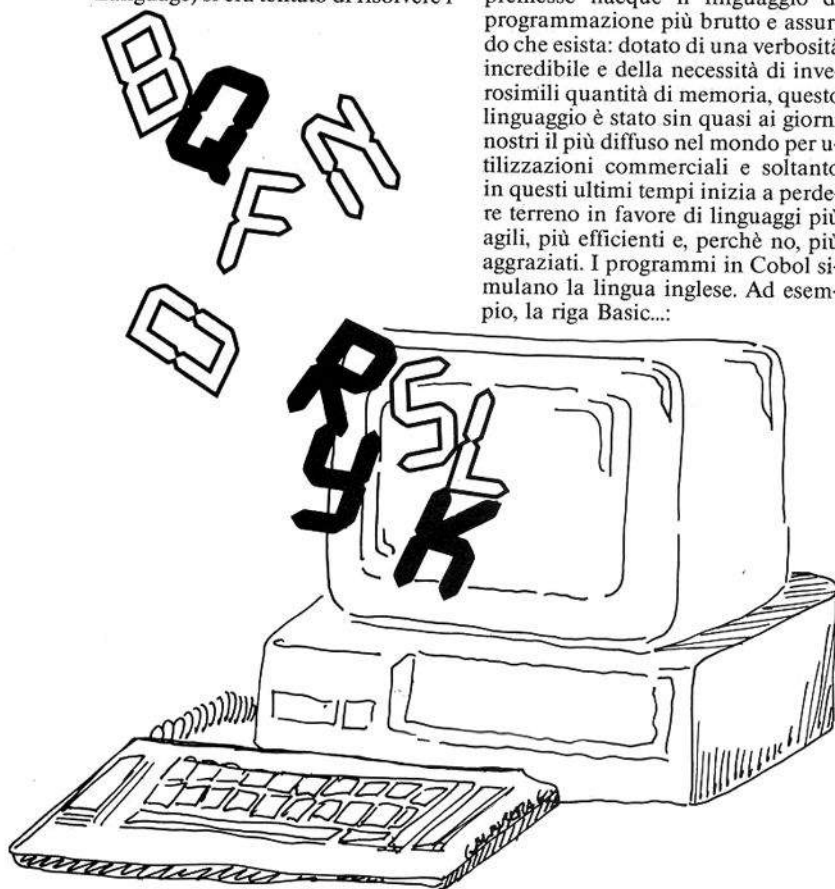
*IF X GREATER THAN ZERO DIVIDE B BY X GIVING A*

In Cobol le funzioni matematiche superiori mancano totalmente ma, quand'anche esistessero, verrebbero inevitabilmente trasformate in mostruosi ginepraі logorroici, incomprensibili ai più. I testi dei programmi assomigliano molto a prosa inglese.

Per esempio, ogni comando viene introdotto da un verbo. Le quattro operazioni non vengono indicate con un segno ma, come si è visto sopra, con comandi del tipo ADD, DIVIDE e così via. Vi sono ancora molti centri di calcolo che lavorano con questo linguaggio, il cui unico vantaggio consiste nella capacità di gestire grandi masse di dati. Se, malgrado ogni avvertimento, voleste studiare questo linguaggio, esso è disponibile sia per il C-64 (fornito di modulo hardware aggiuntivo con Z-80) oppure in CP/M per il C-128.

## Il PL/1

Col passar del tempo anche le imprese commerciali iniziarono ad utilizzare metodi di ricerca statistica, che innanzi erano stati un campo specialistico riservato a pochi eletti. Per soddisfare le esigenze della Ricerca Operativa nei campi statistici ed economici nacque verso la fine, dalla fusione di Fortran, Algol e Cobol (e di un linguaggio pressocchè sconosciuto, il JOVIAL), il PL/1.

Purtroppo però questo linguaggio, sviluppato dalla SHARE (Society for Help to Avert Redundant Effort) in collaborazione con la IBM e implementato quasi esclusivamente su calcolatori della famiglia IBM/360 (che appartenevano alla terza generazione di calcolatori) sembra aver preso la parte peggiore di queste tre lingue: secondo lo stile di chi programma, il programma sembra scritto in Cobol, Fortran o Algol oppure, nella peggiore delle ipotesi, in un miscuglio dei tre. Questo linguaggio non conosce le

parole chiave riservate (cioè comandi riservati all'uso esclusivo del linguaggio), per cui sono possibili costruzioni del tipo

*IF ELSE=THEN THEN IF=ELSE; ELSE IF=THEN*

Il linguaggio, inoltre, in virtù della sua complessità, necessita di tempi decisamente lunghi per l'apprendimento.

I linguaggi "classici" quali il Fortran, il Cobol e l'Algol risentono fortemente delle limitazioni imposte dall'hardware degli anni '50. In quei tempi la capacità di calcolo, e la disponibilità di memoria, erano fattori limitanti, e nessuno pensava di utilizzare i costosi calcolatori per altro che non fosse la gestione di laboriosi calcoli o di grosse masse di dati. Tali linguaggi, pertanto, conoscono soltanto un tipo di dati, vale a dire quelli numerici. Ciò ebbe anche, come conseguenza, il loro utilizzo anche negli anni '60 e '70 in campi per i quali non erano stati creati: vennero scritti in Fortran persino programmi per l'elaborazione di testi, dove ogni singola lettera veniva codificata con un valore numerico e memorizzata in campi numerici.

## Il Pascal

Tra i linguaggi di programmazione "classici" un posto a parte merita il PASCAL. Il nome non è una abbreviazione, ma un omaggio al matematico francese Blaise Pascal che nel 1642, all'età di 19 anni, costruì una macchina per calcolare.

Il linguaggio venne sviluppato dal professore svizzero Nikolaus Wirth nel 1968, e si ricollega ai principi della programmazione strutturata. Si tratta di un linguaggio di singolare eleganza e, a causa della sua costruzione estremamente logica, viene utilizzato per l'insegnamento dell'informatica nelle scuole e università.

Prima di sedersi davanti al computer è necessario analizzare il problema e suddividerlo negli elementi che lo compongono. Poi, per ogni elemento, andrà studiato un algoritmo, che verrà quindi formulato in Pascal.

Tale procedimento, oltre che essere altamente formativo per il futuro programmatore ("costretto" a ragionare, prima di digitare qualsiasi cosa), ha anche il pregio di ridurre al minimo le possibilità di errore che, nel caso di un linguaggio compilato, qual'è il Pascal, comportano un notevole numero di passaggi nel compilatore.

Un programma in Pascal è chiaramente suddivisibile in una parte che contiene le dichiarazioni di tutte le costanti, le variabili e della tipologia dei dati, e in una parte che contiene le procedure necessarie.

Il programma principale, che si trova sempre in fondo al listato, si limita a chiamare le singole procedure. Rispetto al Basic, il Pascal presenta una tipologia di dati molto più estesa, che apre nuovi orizzonti. I dati si suddividono in tipi semplici (integer, real, char, boolean), strutturati (set, array, file, record, ecc.) e in puntatori (pointer). Questi ultimi permettono la concatenazione di liste e la loro manipolazione, nonchè operazioni su strutture ad albero. Il linguaggio ha inoltre il pregio di consentire un rapido apprendimento, per cui è altamente consigliabile a tutti coloro che vogliano approfondire le conoscenze nel ramo dell'informatica. Vi sono parecchie versioni di Pascal che "girano" sul C-64, C-128 e AMIGA.

## Il Basic

Tra il 1956 e il 1971 John Kemeny e Thomas Kurtz del Dartmouth College negli USA svilupparono il linguaggio che è tra i più diffusi e noti al mondo, e che deve essere considerato parte integrante della formazione di base di chiunque si interessi ai computer: il Basic.

Questo è uno dei pochi linguaggi per computer che sia stato creato senza l'intervento di una lobby di potere industriale o militare. Lo scopo dei suoi creatori era quello di offrire agli studenti, che non dovessero necessariamente diventare esperti di informatica o ingegneria elettronica, un modo più facile di programmare di quello a cui si era sino ad allora abituati. Il Basic venne dotato di strumenti adatti al calcolo matematico, ponendo però l'accento sul fatto che doveva essere di facile apprendimento e di ancor più facile impiego. Al contrario di quello che erano state sino ad allora le altre lingue, il Basic venne concepito come un sistema completo e interattivo a sè stante.

L'utente può "dialogare" col Basic nel cosiddetto "modo diretto", che supporta lo "editing", l'esecuzione e il "salvataggio" di programmi senza dover necessariamente uscire dall'ambiente Basic. Questo linguaggio non divenne mai popolare tra i professionisti, soprattutto per ragioni strettamente legate agli interessi dei produttori di computer di allora, e ciò ebbe, come conseguenza diretta, il fatto che il Basic non conobbe mai una standardizzazione ufficiale. Attualmente sono in circolazione parecchie centinaia di "dialetti" Basic, uno diverso dall'altro. Non v'è certamente bisogno di dire che il Basic è il linguaggio implementato su C-64, C-128 e AMIGA!

## Il Forth

All'inizio degli anni '70 Charles H. Moore sviluppo' il linguaggio Forth per il controllo di radiotelescopi. Egli lavorava con un computer IBM 1130, della terza generazione, ma il prodotto finale era talmente potente che il computer che Moore utilizzava sembrava essere della quarta generazione, e per tale ragione Moore volle battezzare il linguaggio "quarto" (in inglese "FOURTH"). Siccome, però, il computer IBM non ammetteva nomi con più di cinque (!) caratteri, il nome divenne FORTH con la caduta della "u". Il Forth è un linguaggio interattivo, come il Basic, vale a dire esiste sia un interprete che un compilatore, il che consente di provare un programma in modo diretto per poi compilarlo. La strutturazione in Forth avviene mediante la definizione di nuove "parole", il che comporta la creazione di un nuovo vocabolario. Le "parole", già create e definite, possono essere inserite in nuove "parole", e così via, tanto che, alla fine della catena, basta un solo comando per far funzionare tutto il programma. Per esempio, digitando i seguenti

# Directory

*Tutti i programmi pubblicati su questo numero di Commodore Computer Club, sono registrati su un dischetto appartenente alla serie "Directory".*

*Oltre ai programmi citati, sono presenti altri file di notevole interesse per coloro che desiderano realmente utilizzare il proprio computer.*

*Sono infatti presenti, di solito, anche i programmi pubblicati sull'altra nostra rivista "Personal Computer", ed altri file che, in totale, riempono quasi per intero i 664 blocchi normalmente disponibili su un floppy disk.*

*Sono disponibili i seguenti dischetti:*

***Directory N. 1** (CCC N.34 + raccolta dell'intera Enciclopedia di routine)*
***Directory N. 2** (CCC N.35 + CCC N.36)*
***Directory N. 3** (CCC N.37 + PC N.7 + PC N.8)*
***Directory N. 4** (CCC N.38 + file vari)*
***Directory N. 5** (CCC N.39 + PC N.9)*
***Directory N. 6** (CCC N.40 + PC N.10)*
***Directory N. 7** (CCC N.41 + file vari)*
***Directory N. 8** (CCC N.42 + file vari)*
***Directory N. 9** (CCC N.43 + file vari)*

## Come procurarsi i dischetti della serie "Directory"

*Avvertiamo i lettori che NON è assolutamente possibile inviare i programmi su nastro, per intuibili motivi di economia ed affidabilità del nastro cassetta.*

*Ogni numero di "Directory" può quindi esser richiesto SOLO su disco inviando L.12000 per ciascun disco oltre a L.3000 (fisse) per le spese di imballo e spedizione (indipendenti dal numero di dischi richiesti).*

*Chi desiderasse la spedizione raccomandata, deve aggiungere altre 3000 lire per l'ulteriore affrancatura.*

*Non ci è possibile inviare materiale contrassegno: si prega di astenersi dal chiedere eccezioni alla regola.*

*Compilate un normale modulo di C/C postale indirizzando a:*

***C/C postale N. 37952207***
***Systems Editoriale***
***Viale Famagosta, 75***
***20142 Milano***

*Non dimenticate di indicare chiaramente, sul retro del modulo (nello spazio indicato con "Causale del versamento") non solo il vostro nominativo completo, ma anche il numero del disco desiderato; esempio:*

***"Directory N.1"***
***"Directory N.3"***
***"Directory N.4"***

***Totale:***
***L.12000x3 +***
***L.6000 (spediz. racc.)***
***= L.42000***

*(spese di imballo e spediz. racc. comprese).*

*N.B. Per ottenere il materiale ordinato in tempi più ristretti, inviate l'importo a mezzo assegno bancario non trasferibile con lettera di accompagnamento: le poste italiane non brillano per velocità! (due mesi circa per il recapito di un C/C postale).*

comandi in modo diretto, possiamo costruire una frase:

*: A1 ." I LOVE " ; [RETURN]*
*: A2 ." MY " ; [RETURN]*
*: A3 ." COMPUTER " ;[[RETURN]*
*: B1 A1 A2 A3 ; [RETURN]*

Digitando B1, seguito da return, vediamo che il computer stamperà sullo schermo: "I LOVE MY COMPUTER OK". Come si vede è possibile definire nuove parole inglobandone altre, con evidenti vantaggi.

Il Forth si basa, in modo abbastanza insolito, su di una memoria detta STACK (pila, catasta) che funziona in base al principio LIFO (Last In First Out, ovverossia "l'ultimo [dato] immesso, è il primo [dato] a uscire"). Ciò comporta il fatto che tutti i valori immagazzinati nello Stack possono essere "ripresi" soltanto in senso inverso a quello con cui sono stati immagazzinati. Il Forth utilizza, inoltre, la Notazione Polacca Inversa, abbastanza nota a chi abbia usato i calcolatori scientifici della Hewlett-Packard.

## Il Logo

Sulla spinta delle teorie sull'apprendimento del filosofo svizzero Jean Piaget, Seymour Papert del Massachusetts Institute of Technology lavorò per circa 12 anni allo sviluppo di un linguaggio per computer. Nacque, così, il LOGO che, molto meglio del Pascal o del Basic, si adatta al modo di pensare dello studente. Nel Logo è implementato il concetto basilare del LISP (un altro linguaggio poco conosciuto, il cui nome è una abbreviazione di LISt Programming). Le liste sono facili da definire e possono essere manipolate, comparate e combinate per mezzo di opportuni comandi. Purtroppo, però, il Logo necessita di grossi quantità di memoria per cui, almeno per il momento, è implementato sui microcalcolatori soltanto per ciò che riguarda la sua parte più spettacolare, vale a dire la "tartaruga". Questo è il nome con cui è più nota la "turtle-graphics", per mezzo della quale è possibile tracciare sullo schermo bellissime pagine grafiche in modo molto semplice. Uno dei più importanti concetti del Logo è la ricorsività: le procedure sono in grado di richiamare se stesse. Ricordiamo che le strutture ricorsive non esistono in Basic nè in molti altri linguaggi e quei pochi linguaggi che sono in grado di implementarle possono farlo soltanto a fatica.

## Il Comal

Nel 1973 venne sviluppato il linguaggio COMAL che deriva dall'unione di Basic e Pascal, con elementi di Logo. Nel Comal il compilatore e l'interprete non sono elementi separati, ma integrati in un qualcosa di intermedio. Un programma in Comal si compone di tre fasi. In una pri-

ma fase, che ha luogo durante l'introduzione del programma, ne viene controllata la sintassi, e i comandi del Comal vengono immediatamente tramutati nei cosiddetti "token" (tutti gli interpreti utilizzano lo stesso principio, vale a dire tramutano una parola-comando, lunga parecchi byte, in un comando della lunghezza di un solo byte). La seconda fase ha inizio nel momento in cui il programma viene fatto girare: il testo del programma viene analizzato e le variabili, le procedure, le funzioni e i salti vengono raccolti in una sorta di lista automatica che viene messa da parte. La terza fase è il funzionamento vero e proprio: l'interprete fa ricorso a quella lista senza dover cercare per tutto il programma quello di cui ha bisogno, e ciò comporta un enorme aumento della velocità di esecuzione del programma. Il Comal si rifà, per quanto riguarda le sintassi, alle regole del Basic, ma ha preso dal Pascal il principio della strutturazione senza per questo dover ricorrere alla rigorosa sintassi del Pascal.
Esistono bellissime versioni del Comal sia per il C-64 che per il C-128.

## Il "C"

Nel frattempo un nuovo linguaggio, nato nell'ambito del sistema operativo Unix a 16 bit, è andato via via affermandosi in questi ultimi tempi e vi è un sempre maggior numero di programmatori che lo indica come il linguaggio di loro gradimento. Si tratta del linguaggio noto come "C". Il nome è dovuto al fatto che si stava cercando di sviluppare una lingua speciale per i microcomputer della Digital Equipments, e le varie versioni venivano semplicemente designate con lettere dell'alfabeto. Un programmatore di nome Dennis Ritchie della Bells Laboratories, nel 1971 riconobbe le possibilità implicite nel linguaggio denominato "B", lo modificò, e la sua versione venne chiamata "C". Poco più tardi, nel 1973, il sistema operativo Unix venne riscritto in "C". Per la prima volta nella storia dei computer veniva impiegato un linguaggio di "alto" livello, anzichè l'Assembly, per scrivere un sistema operativo. Ciò può dar la misura dell'essenzialità e economicità del linguaggio. La programmazione in "C" si basa su una accentuata strutturazione, sul modello del Pascal, che porta ad una estrema efficienza e compattezza. D'altro canto, però, il "C" possiede una serie di operatori e comandi simili a quelli dell'Assembly per il migliore utilizzo delle possibilità del processore. Il linguaggio "C" ha un numero molto limitato di comandi implementati, il che comporta che ogni programmatore è costretto a scrivere da sè le routine che interessano. Nonostante la sua apparente semplicità, il linguaggio "C" necessita purtuttavia di un lungo periodo di apprendimento: al fine di ottenere la massima flessibilità di utilizzo, generalmente non vengono visualizzati messaggi di errore. Malgrado alcuni svantaggi, il linguaggio è particolarmente efficiente e, a dimostrazione di ciò, ricordiamo che il sistema operativo del nuovo Commodore "AMIGA" è stato scritto quasi completamente in "C". Esiste una versione ridotta del "C" sia per il C-64 che per il C-128, ma la versione migliore è il sistema di programmazione in "C" studiato dalla Lattice per l'AMIGA.

## Modula

Il concetto che permea i moderni linguaggi per computer è la strutturazione del "problem solving" in moduli, vale a dire compiti parziali del programma vengono risolti in programmi più piccoli, che vengono detti "moduli". I vantaggi di un tal modo di procedere sono evidenti: un problema di programmazione deve essere risolto una volta soltanto. Se lo stesso problema dovesse ripresentarsi, sarà sufficiente riutilizzare il modulo precedentemente impiegato. Inoltre è certamente più semplice suddividere un programma in una serie di programmi più piccoli, semprechè il linguaggio impiegato lo consenta. Il "linguaggio modularizzato" permette dunque la razionalizzazione dei programmi. Il Prof. Nikolaus Wirth, già creatore del Pascal, ha colpito ancora col nuovo linguaggio MODULA. Ogni programma scritto in Modula è un programma a sè con dati, funzioni e variabili suoi propri, ma può "chiamare" altri programmi-moduli che a loro volta utilizzano dati, funzioni e variabili che sono loro propri. Il linguaggio è ancora poco noto e non sembra essere stato ancora implementato su piccoli calcolatori.

## Ada

C'è chi sostiene che attualmente il "massimo" sia rappresentato dal linguaggio "Ada". Il nome del linguaggio deriva da quello della contessa Ada Lovelace, figlia del poeta inglese Lord Byron. La contessa visse in Inghilterra nel XVIII secolo ed è la prima persona che mise a punto un programma per la "macchina per calcolare" di Charles Babbage, citato all'inizio di questo articolo. Sinora ADA era conosciuto soltanto a pochi iniziati che lavoravano sui mainframe, ma recentemente sono state messe a punto versioni per computer che, come dice la pubblicità americana, "people like you and I own" (cioè "computer del tipo che gente come voi e come me possiede"). Il linguaggio Ada venne concepito su incarico del Ministero statunitense della Difesa e anch'esso, come il "C" e il Modula, si basa sul concetto del Pascal. Tutte le operazioni di Input/Output non appartengono al linguaggio, ma vengono chiamate da moduli che, in caso di necessità, possono essere riscritti dal programmatore stesso. Ada rende possibili le cosiddette EXCEPTIONs (situazioni eccezionali nel programma, soprattutto situazioni di errore) senza perdere mai il controllo della macchina, con moduli simili ai comandi Basic "ON ERROR GOTO" oppure "TRAP". Inoltre ADA consente il cosiddetto "multitasking" (naturalmente su computer che lo consentono), vale a dire la possibilità di portare avanti certe operazioni in contemporanea, il che comporta un enorme aumento della velocità di esecuzione. La Data Becker tedesca e la Abacus Soft americana hanno immesso sul mercato un "ADA Training Course" per C-64 e C-128 il quale, benchè limitato, dà la possibilità di farsi una idea precisa delle possibilità e dei limiti di questo linguaggio.

# Espansione gratuita di memoria Ram

*Come incrementare di quasi mille byte la memoria a disposizione del C/16*

**di Roberto Ferro**

**SCHEDA TECNICA**

Note didattiche applicative di utilità.

Anche il programma pubblicato in queste pagine è contenuto nel disco "Directory" di questo mese.

Programma specifico per C/16, pur se applicabile (dopo opportune modifiche) ad altri computer Commodore.

Consigliato per applicazioni grafiche in Basic.

Come tutti sanno la quantità di memoria RAM disponibile sul C/16, da Basic, è di 12K Ram che, tutto sommato, risulta più che sufficiente per la stesura dei programmi che siamo soliti scrivere e utilizzare.

I problemi cominciano quando si ricorre alle istruzioni grafiche del piccolo computer, dal momento che occorrono 8K per "entrare" in alta risoluzione, e 2K per gli attributi di colore e luminanza: dei 12K disponibili non ne rimangono che un paio.

Il (micro) programma proposto cerca proprio di ovviare, per quanto possibile, all'inconveniente, sfruttando una tecnica in parte accennata nell'inserto del N.44. Estendendo la zona del Basic fin sull'area video, è infatti possibile aumentare la RAM di quest'ultimo di circa un kilobyte.

Attivata l'espansione, sarà possibile, ovviamente, utilizzare solo la porzione di video superiore per manipolare i vari programmi. Bisognerà quindi prestare la MASSIMA attenzione a NON "uscire" dalla finestra delimitata: ciò sarebbe possibile, come è noto, impartendo inopportuni comandi di Window, attivabili servendosi del carattere Esc [Chr$(27)], oppure premendo due volte di seguito il tasto Home.

L'invasione che il Basic compie ai danni del video è attuata con l'ausilio di alcune POKE che cambiano, alle locazioni 43-50 (esadecimale $ 2B-32), i puntatori di inizio Basic.

## Il programma

Il listato estremamente breve svolge tre funzioni:

- Definisce una finestra video che durante l'uso dell'espansione non deve, come già detto, essere cancellata (pena la perdita del programma correntemente in memoria e un probabile inchiodamento del sistema).

- Immette nella memoria colore, relativa ai caratteri, lo stesso colore dello sfondo.

- POKa nelle locazioni dei puntatori dell'inizio testo Basic i nuovi valori.

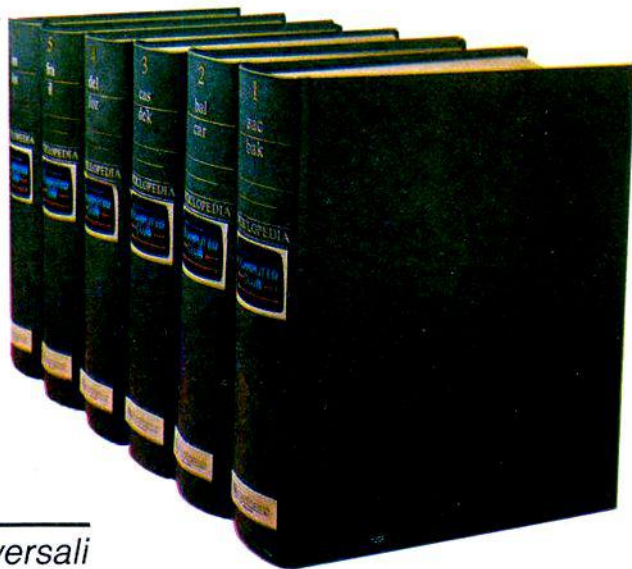
Dopo il RUN bisognerà attendere alcuni secondi affinchè vengano allocati i valori come sopra descritto.

Alla comparsa del consueto Ready potremo chiedere la quantità di memoria libera ottenendo in risposta un rassicurante 13059 bytes free.

A questo punto ogni cosa sarà come prima, con la differenza di aver a disposizione una maggiore quantità di memoria.

```
10 PRINT"[CLR]"CHR$(27)CHR$(84),,,,"[RVS][DOWN][DOWN][DOWN][DOWN][DOWN]"CHR$(27)CHR$(66)
20 FORI=2288TO3071:POKEI,PEEK(65301):POKEI+1024,0:NEXT
30 POKE43,241:POKE44,12:POKE45,243:POKE46,12:POKE47,243:POKE48,12:POKE49,243:POK
E50,12

READY.
```

# Cinque routine per tutti i gusti

*Quattro sottoprogrammi Basic universali (più uno specifico per C/64) utili per potenziare i vostri programmi*

## 18300 String-player
**(Commodore 64)**

Il comando Play del C-128 è molto comodo perchè rende possibile, mediante una stringa contenente le note (in notazione anglosassone), ottenere l'esecuzione del brano.

E' evidente che tale sistema è molto comodo e facilita di molto il compito dell'utente.

I possessori del C/64 sanno, peraltro, che non è così semplice ottenere suoni e musica dal loro computer.

La routine proposta permette quindi di concatenare una stringa di note e di farla suonare dal SID.

A causa delle "limitazioni" derivanti dallo standard dell'Enciclopedia, la routine non simula il comando Play in tutte le sue (complesse) forme sintattiche; è in grado infatti di suonare soltanto sette note e, per di più, senza alterazioni.

Nonostante ciò, la modularità con cui la routine è stata costruita rende possibili notevoli ampliamenti (che in questa sede non sarebbero stati realizzabili).

Ricordiamo, a chi non la conosce ancora, la corri-

spondenza fra le note "italiane" e quelle "anglosassoni"

- DO = C
- RE = D
- MI = E
- FA = F
- SOL= G
- LA = A
- SI = B

Per far suonare la voce 1 del SID è necessario settare alcuni registri che svolgono particolari funzioni:

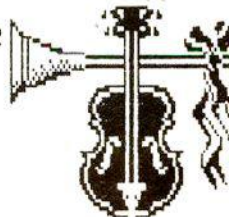
- 54273 = Frequenza (Hi byte)
- 54272 = Frequenza (Lo byte)
- 54296 = Volume
- 54276 = Forma d'onda
- 54277 = Attack/Decay

Per far suonare le note è opportuno conoscere i valori da "pokare" nei byte di Frequenza (54273 / 54272); nell'appendice del manuale del C/64 è riportata una tabella esplicativa che ripresentiamo (in parte) per motivi di comodità:

| Nota | High | Low |
|---|---|---|
| A (La) | 14 | 107 |
| B (Si) | 16 | 47 |
| C (Do) | 1 | 37 |
| D (Re) | 19 | 63 |
| E (Mi) | 2 | 154 |
| F (Fa) | 2 | 227 |
| G (Sol) | 2 | 177 |

E' possibile ottenere un ritardo nell'esecuzione servendosi della variabile X4 (linea 18335). Per far suonare la stringa X0$ è quindi sufficiente effettuare un Gosub 18300.

Le linee 18310 - 18325 controllano la legalità della nota (pena l'emissione di una segnalazione di errore). Tramite la funzione ON... GOTO, vengono settati i byte di frequenza con il valore che corrisponde alla nota; questa viene eseguita ed il nome (se X6 è maggiore di zero) viene visualizzato sullo schermo.

```
  100 REM DEMO STRING-PLAYER
  110 REM BY MICHELE MAGGI
  120 REM SUONA LE NOTE CONTENUTE
       IN X0$
  130 REM (SOLO PER C/64)
  140 :
  150 X4=150:X6=1:REM  SETTA TEMP
      O + ABILITA PRINT
  155 X0$="EEFGGFEDCCDEEDD":GOSUB
       18300
  160 X0$="EEFGGFEDCCDEDCC":GOSUB
       18300
  170 PRINT:INPUT "DIGITA STRINGA
      ";X0$:GOSUB 18300
  180 IF X0$="ERR" THEN PRINT"ERR
      .IN STRINGA"
 9998 :
 9999 END
18300 POKE 54296,15:X2=54273:X3=5
      4272
18310 FOR  X1=1  TO  LEN (X0$)
18320 X1$=MID$(X0$,X1,1):XX=ASC(X
      1$)-64
18323 IF XX<1 OR XX>7 THEN PRINT:
      X0$="ERR":RETURN
18325 ON  XX GOTO  18350,18355,18
      360,18365,18370,18375,18380
18330 POKE 54277,9:POKE 54276,17:
      IF X6>0 THEN PRINTX1$;
18335 FOR X5=1 TO X4:NEXT:POKE 54
      276,0:NEXT
18340 POKE X2,0:POKE X3,0
18345 RETURN
18349 :
18350 POKE X2,14:POKE X3,107:GOTO
        18330
18355 POKE X2,16:POKE X3,47:GOTO
       18330
18360 POKE X2,17:POKE X3,37:GOTO
       18330
18365 POKE X2,19:POKE X3,63:GOTO
       18330
18370 POKE X2,21:POKE X3,154:GOTO
        18330
18375 POKE X2,22:POKE X3,227:GOTO
        18330
18380 POKE X2,25:POKE X3,177:GOTO
        18330
18389 REM  STRING-PLAYER
```

## 18400 Password
**(Qualsiasi Commodore)**

Computer e segretezza: tantissimi sistemi computerizzati richiedono, per consentire l'accesso all'utente, una o più parole-chiave (Bancomat ecc. ecc.).
Vediamo ora come è possibile simulare un "Password Construction Set" in modo da permettere l'utilizzo di un programma solo ad utenti autorizzati.
Il funzionamento della subroutine, molto semplice, richiede due parametri:

- La password, da assegnare alla stringa X0$.
- Il numero massimo di tentativi permessi, da assegnare alla variabile XX.

Ogni volta che verrà premuto un tasto sarà visualizzato un asterisco (per favorire la segretezza della digitazione nel caso in cui l'utente sia costretto ad operare in presenza di estranei); solo alla pressione del tasto Return verrà effettuato il controllo della password.
In caso positivo sarà concesso il controllo del programma principale; in caso contrario verrà incrementato il contatore X0 e, qualora fosse maggiore del numero massimo di tentatvi XX, si avrà l'interruzione del programma.
I più cattivi potranno escogitare vendette tremende: reset del sistema, valanghe di insulti, formattazione del dischetto inserito, sfregio permanente, bollo falso e così via...
La password non ha particolari problemi di lunghezza e può essere formata anche dai "caratteri speciali" tipo Cursor up/down, Delete e così via.
Va da sè che se la password contiene molti caratteri speciali sarà molto più difficile da indovinare. Un consiglio: prima di compilare o tradurre questo programma in L.M. annotate la password da qualche parte!

```
  100 REM PASSWORD
  110 REM BY MICHELE MAGGI
  120 REM PER QUALSIASI COMMODORE
  130 :
  150 X0$="PIPPO":XX=3:GOSUB 1840
      0
  160 IF X0>XX THEN PRINT"IL PROG
      RAMMA TERMINA!":END
  170 PRINT"IL PROGRAMMA CONTINUA
      ...."
 9998 :
 9999 END
18400 PRINT">";:X1$="":X2$=""
18405 GET X1$:IF X1$="" THEN 1840
      5
18410 IF X1$=CHR$(13) THEN 18450
18420 PRINT"*";:X2$=X2$+X1$
18430 GOTO 18405
18450 IF X2$=X0$ THEN 18470
18460 X0=X0+1:IF X0>XX THEN PRINT
      :PRINT"ACCESSO NEGATO":RETU
      RN
18465 PRINT:PRINT"TENTATIVO"X0" E
      RRATO":GOTO 18400
18470 PRINT:PRINT"ACCESSO CONSENT
      ITO!":RETURN
18489 REM PASSWORD
```

## 18500 String-Writer
**(Qualsiasi Commodore)**

A volte, dovendo visualizzare un testo, è meglio presentarlo lentamente, lettera per lettera, piuttosto che in un blocco unico.
E' infatti certo che chi legge è più stimolato a seguire un testo, per così dire, "in formazione" piuttosto che un testo stampato in un blocco unico che rischia di essere più pesante e faticoso da digerire.
La routine proposta permette, appunto, di visualizzare un testo (contenuto in X0$) con un ritardo programmabile (contenuto in X3).
Dopo aver settato il ritardo e digitato il testo sarà sufficiente chiamare la routine per vederlo visualizzato più o meno lentamente in funzione del valore assegnato a X3; premendo il tasto Return, durante la visualizzazione, questa viene sospesa. Premendo un tasto qualsiasi, riprende.

```
  100 REM STRING-WRITER
  110 REM BY MICHELE MAGGI
  120 REM PER QUALSIASI COMMODORE
  130 :
  150 X3=100:REM RITARDO
  155 X0$="COMMODORE COMPUTER CLU
      B":GOSUB 18500
  160 X3=20:X0$="TI INSEGNA A PRO
      GRAMMARE":GOSUB 18500
 9998 :
 9999 END
18500 FOR X0=1 TO LEN(X0$)
18515 GET X1$:IF X1$=CHR$(13) THE
      N GOSUB 18560
18520 PRINTMID$(X0$,X0,1);
18530 FOR X2=1 TO X3:NEXT
18540 NEXT
18550 PRINT:RETURN
18560 FOR X4=1 TO 100:NEXT
18562 GET X1$:IF X1$="" THEN 1856
      2
18563 RETURN
18599 REM STRING WRITER
```

## 18600 Incornicia risultati
**(Qualsiasi Commodore)**

Dovendo presentare il risultato di un calcolo, non importa se semplice o complesso, può essere utile "vestirlo" in modo da renderne più simpatico l'output.

La routine 18600 svolge appunto questo lavoro; tramite i caratteri semigrafici Commodore crea una cornice attorno al risultato.

Potrà sembrare strano che nel caso di un risultato positivo (maggiore di zero) venga stampato uno spazio prima del numero; ciò dipende dal fatto che il computer stampa anche il segno che, in questo caso, essendo positivo, resta sottinteso. Provate a far incorniciare un risultato negativo e ne avrete la conferma.

Il numero da incorniciare deve essere contenuto in X0 e la stringa X0$ deve esser nulla (vedi riga 150).

Se si accede alla routine con la stringa X0$ non nulla (vedi riga 155) ad essere incorniciata sarà quest'ultima, purchè la sua lunghezza sia lunga, al massimo, 37 caratteri, per ovvi motivi di "ospitalità" dello schermo.

```
  100 REM INCORNICIA I RISULTATI
  110 REM BY MICHELE MAGGI
  120 REM PER QUALSIASI COMMODORE
  140 :
  150 X0=10*(70-10):X0$="":GOSUB
      18600
  151 GET A$:IF A$="" THEN 151
  155 X0$="QUESTO E' UN MESSAGGIO
       LUNGO 37 CARAT":GOSUB 1860
      0
  160 PRINT:INPUT "DIGITA XO$";X0
      $:GOSUB 18600
  170 IF X0$="ERR" THEN PRINTX0$
 9998 :
 9999 END
18600 IF X0$="" THEN 18603
18601 IF LEN(X0$)<38 THEN PRINTCH
      R$(142):GOTO 18605
18602 X0$="ERR":RETURN
18603 X0$=STR$(X0):PRINTCHR$(142)
18605 PRINT"╭";
18610 FOR X1=1 TO LEN(X0$):PRINT"
      —";:NEXT
18615 PRINT"╮";
18620 PRINT:PRINT"|"X0$"|"
18625 PRINT"╰";
18630 FOR X1=1 TO LEN(X0$):PRINT"
      —";:NEXT
18635 PRINT"╯";
18640 RETURN
18699 REM INCORNICIA RISULTATI
```

## 18700 Directory segreta
**(Qualsiasi Commodore + Drive)**

Ecco una routine, dedicata ai "maniaci" delle protezioni, con cui è possibile formattare un dischetto sia normalmente, che col metodo di formattazione veloce (senza ID) in modo che caricando successivamente la directory con...

LOAD "$",8

...sia possibile listare solo il nome del disco invece che l'intero suo contenuto.

Per ciò che riguarda le procedure di caricamento e salvataggio dei programmi, la procedura è quella solita; sarà quindi possibile salvare e caricare nel modo consueto, con l'unica differenza che i nomi dei programmi non appariranno nella directory (morale: scrivete a parte, su un foglio di carta, i nomi dei file che memorizzerete nel dischetto così conciato).

Il funzionamento della routine è molto semplice: il nome del disco viene fatto precedere da tre codici Delete, CHR$(20) ed è seguito da tre CHR$(0).

Ciò ingannerà l'interprete Basic che, al momento di listare la directory, ometterà di visualizzare i primi tre caratteri del nome del disco che, in genere, sono lo zero (0), lo spazio e gli apici (").

Con la routine proposta, l'interprete, dopo aver stampato in reverse il nome del disco, trovando tre zeri consecutivi crederà di essere arrivato alla fine del programma e stamperà Ready restituendo il controllo all'utente.

Non dimentichiamo, infatti, che una directory caricata con LOAD "$",8 viene trattata come se fosse un programma Basic, anche se non eseguibile.

L'unica limitazione deriva dal fatto che un disco formattato in questo modo può avere un nome lungo al massimo dieci caratteri in quanto gli altri sei sono "occupati" dai tre Delete e dai tre zeri.

E' ovvio che se si esamina la directory con una routine apposita verranno visualizzati ugualmente i nomi dei programmi presenti, a dispetto della routine utilizzata.

Per motivi di sicurezza, prima di passare alla formattazione, viene chiesta la pressione del tasto "S" per la conferma, in modo da evitare accidentali perdite di dati preziosi.

```
100 REM DIRECTORY SEGRETA
110 REM BY MICHELE MAGGI
120 REM PER QUALSIASI
130 REM COMMODORE + DRIVE
140 :
150 INPUT "NOME DISCO,ID";X0$,X
    1$
160 IF X0$="" THEN 150
170 GOSUB 18700
```

```
 9998 :
 9999 END
18700 Z1$=CHR$(20)+CHR$(20)+CHR$(
      20):Z2$=CHR$(0)+CHR$(0)+CHR
      $(0)
18705 IF LEN(X0$)>10  THEN  PRINT
      "ERR":END
18710 IF LEN(X1$)>2  THEN  PRINT"
      ERR":END
18720 PRINT"CONFERMI FORMATTAZION
      E?"
18730 GET X5$:IF X5$="" THEN 1873
      0
18740 IF  X5$="S"  THEN 18750
18745 PRINT"ESECUZIONE INTERROTTA
      ":END
18750 OPEN 1,8,15,"N:"+Z1$+X0$+Z2
      $+","+X1$:CLOSE 1
18760 PRINT"FORMATTAZIONE COMPLET
      ATA"
18770 RETURN
18799 REM DIRECTORY SEGRETA
```

**Elenco delle ultime routine pubblicate**

(Fra parentesi è riportato il numero di Commodore Computer Club su cui sono apparse)

# Uno schermo in frantumi

*Una interessante applicazione in L.M. per creare simpatiche schermate di presentazione*

di Michele Maggi

Spesso si sente dire che la prima impressione è quella che conta ed effettivamente, anche in campo informatico, questa massima ha una sua validità.

Qualsiasi software, professionale e non, se corredato di un presentazione insolita potrà dare una buona impressione all'utente per cui molto spesso, a dispetto di un programma mediocre, vediamo presentazioni strabilianti che invogliano il potenziale acquirente.

Quanto detto vale (e come no?) anche per i sessantaquattristi che vogliano dotare i loro programmi di una veste più "professionale".

L'idea è venuta in mente lavorando con il programma Printshop (una diffusissima utility) recensito sul N. 34 di C.C.C.

"Screen Magic", una tra le tante opzioni del programma, permette di creare una scritta a caratteri cubitali su video per poi salvarla o stamparla.

Ci siamo quindi chiesti in che modo fosse possibile utilizzare le schermate ottenute con Screen Magic, in modo anche indipendente da Printshop, per creare presentazioni personalizzate per i nostri programmi.

**SCHEDA TECNICA**

Software applicativo per presentazioni grafiche di notevole effetto.

Idoneo per computer C/64 e non adattabile ad altri computer Commodore.

Richiede la possibilità di utilizzare le schermate grafiche generabili con il package "Printshop".

Consigliato a coloro che intendono acquisire esperienza nel campo dell'Assembly.

Facilmente utilizzabile anche dai principianti.

Anche il programma pubblicato in queste pagine (oltre al alcuni file applicativi) è contenuto nel disco "Directory" di questo mese.

## I primi tentativi

Dopo aver creato e salvato una schermata tramite Screen Magic, la prima cosa da fare è vedere dove Printshop alloca le schermate in questione; ricorrendo ad un programma di Monitor (Zoom & simili) si scopre che il file-schermata salvato su disco è allocato da $4000 a $5F80 (16384-24448).

Nel programma presentato in queste pagine, per motivi di semplicità, carichiamo invece la schermata a partire da $2000 (8192) invece che da $4000 dal momento che questa rappresenta una zona di memoria "classica" per allocare schermate in alta risoluzione.

Per far ciò è sufficiente alterare i puntatori di inizio Basic in modo da dirottarli a 8192 e, successivamente, caricare la schermata:

Poke 43,0: Poke 44,32: Load "filename",8

E' molto importante ricordarsi di utilizzare la semplice forma sintattica ",8" che non va assolutamente sostituita con ",8,1" in quanto non si otterrebbe il caricamento a $2000, ma all'indirizzo originale.

A questo punto digiteremo, in modo diretto, la seguente linea:

Poke 53265,59: Poke 53272,28

Sorpresa! La schermata di Printshop è ora visibile...

Per tornare in modo testo è sufficiente premere Run/Stop e Restore.

Questa prima esperienza rischia, però, di deluderci in quanto i colori della scritta risultano confusi perchè gli eventuali caratteri presenti sullo schermo (ora semplici quadratini colorati) formano gli "attributi colore" della schermata.

Per colorare la scritta sarà necessario riempire i 1000 byte dello schermo con il codice ASCII del colore desiderato (vedi disassemblato linee 38-45).

Prima di procedere sarà opportuno salvare la schermata su disco nel modo seguente:

Poke 45,128: Poke 46,63: Save "schermata",8

Su disco sarà quindi presente il file "Schermata" caricabile (questo è molto importante) con ",8,1" senza alterare i puntatori.

## Digitiamo!

E' finalmente giunta l'ora di digitare (e salvare!) il programma pubblicato.

Una volta fatto tutto ciò sarà possibile caricare la schermata con il suffisso ",8,1" unitamente al programma, nel caso non sia già in memoria.

Una volta dato il RUN, dopo la lettura dei DATA si trasferirà il controllo alla routine in L.M. allocata da $4000 in poi.

Il risultato sarà la visualizzazione della nostra schermata con una specie di pulsazione data da un periodico cambio di colori sulla scala dei tre grigi, oltre al bianco.

Per questa gestione è stato necessario ricorrere al registro di Raster ($D012) in modo da cambiare il colore solo quando il "pennello elettronico" è fuori dal quadro visibile per evitare fastidiosi "sfarfallamenti".

Non appena verrà premuto un qualsiasi tasto, il controllo passerà ad una subroutine che, tramite la generazione di numeri casuali tra 8192 e 16256 (utilizzando l'oscillatore 3), riprodurrà l'effetto di "frantumazione" dello schermo, fino a farlo scomparire del tutto.

Inutile dire che questa caratteristica è quella che fa di questo programmino qualcosa di originale e che vale la pena di provare.

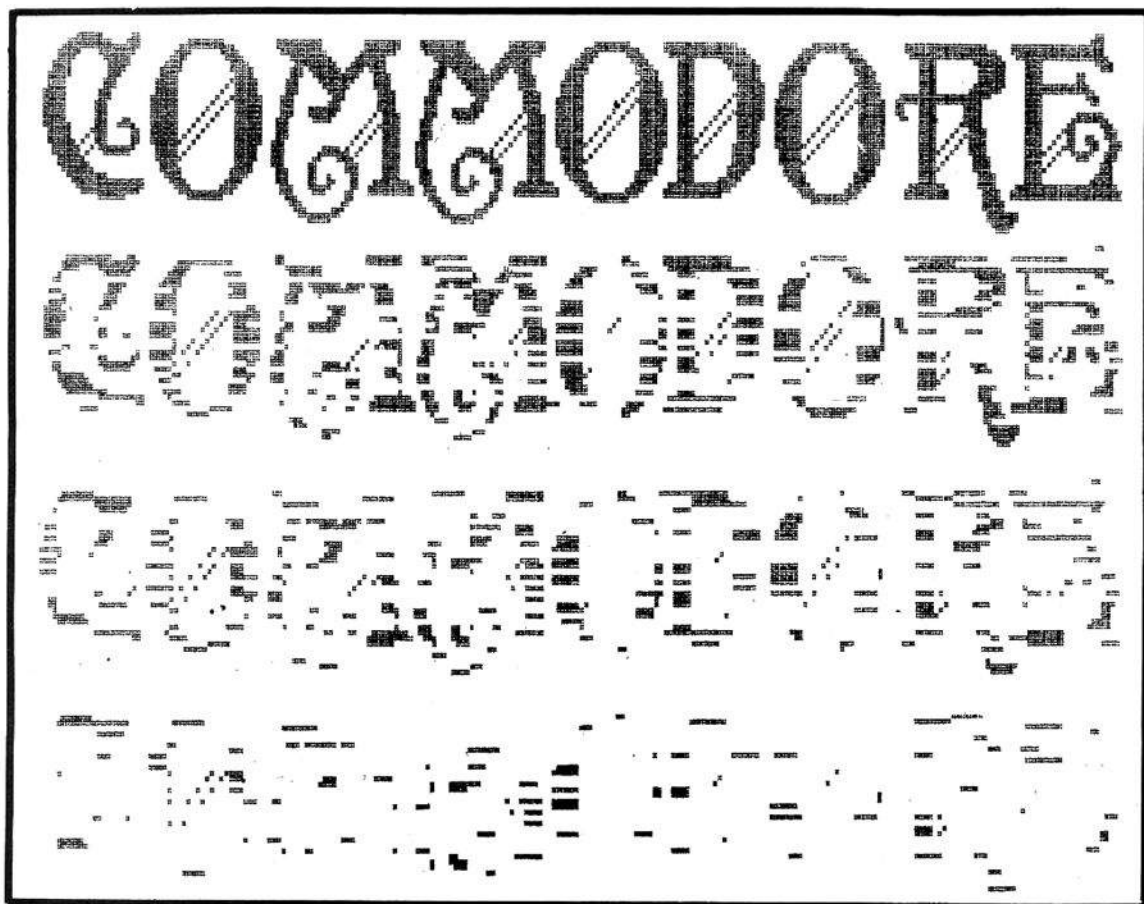
Terminata la routine si ritorna normalmente al Basic.

A questo punto, se si desidera rivedere la schermata sarà necessario ricaricarla da disco.

## Miglioriamo!

Così come è pubblicata, la routine si presta a notevoli miglioramenti (sprite, musica ed altro) che sicuramente i più bravi saranno in grado di realizzare.

Chi fosse interessato a vedere alcune "variazioni sul tema" di questo programma (ovviamente in versione più complessa) può procurarsi il numero 15 di Software Club oppure uno qualsiasi dei numeri di "Commodore 64 Club" in cui le copertine so-

no, appunto, realizzate con sistemi ispirati a quello proposto.

Il disassemblato commentato non necessita di ulteriori spiegazioni ed è facilmente analizzabile anche da coloro che iniziano a masticare l'Assembly.

Su Directory di questo mese è presente, a titolo dimostrativo, oltre al programma in questione, anche una schermata di Printshop (già allocata a $2000) in modo che anche chi non possiede (per il momento) Printshop possa rendersi conto degli interessanti effetti che si possano ottenere.

```
1  *-----------------------*
2  * FRANTUMA-VIDEO PER C/64 *
3  *                       *
4  *  UTILIZZA SCHERMATE DI *
5  *       PRINT SHOP      *
6  *                       *
7  *    BY MICHELE MAGGI   *
8  *                       *
9  *       (C) 1987        *
10 *                       *
11 *-----------------------*
12 *
13          ORG   $4000     ;
14
15          LDA   #147      ;PULISCE
16          JSR   $FFD2     ;LO SCHERMO
17          LDA   #59       ;E SETTA
18          STA   53265     ;LA PAGINA
19          LDA   #24       ;GRAFICA
20          STA   53272     ;A $2000
21 MAIN     NOP             ;
22          LDA   #255      ;PREDISPONE
23          STA   $02       ;L'INDIRIZZAMENTO
24 LP1      INC   $02       ;INDICIZZATO
25          LDA   $02       ;PER I COLORI
26          CMP   #6        ;DELLA
27          BEQ   MAIN      ;SCRITTA
28          LDX   $02       ;
29          LDA   TABLE,X   ;
30          STA   TEMP      ;
31          LDX   #155      ;GENERA
32 RETARD   JSR   $EEB3     ;UN RITARDO
33          DEX             ;DI 155
34          BNE   RETARD    ;MILLISECONDI
35 RAS      LDA   $D012     ;CONTROLLA
36          CMP   #$FF      ;CHE IL RASTER
37          BNE   RAS       ;SIA FUORI
38          LDA   TEMP      ;SCHERMO
39          LDX   #0        ;E CAMBIA
40 LP       STA   $0400,X   ;IL COLORE
41          STA   $0500,X   ;ALLA
42          STA   $0600,X   ;SCHERMATA
43          STA   $06E8,X   ;
44          INX             ;
45          BNE   LP        ;
46          LDA   $C5       ;SE SI PREME
47          CMP   #64       ;UN TASTO
48          BNE   EXIT      ;ESCE
49          JMP   LP1       ;
50 EXIT     NOP             ;SCHERMO IN NERO
51          LDA   #0        ;E SETTA FREQ.
52          STA   53280     ;OSCILLATORE 3
53          STA   54286     ;PER VALORE
54          LDA   #151      ;CASUALE
55          STA   54287     ;
56          LDA   #129      ;ACCENDE
57          STA   54290     ;VOCE 3
58          LDA   #0        ;AZZERA
59          STA   161       ;TI$
60          STA   162       ;
61 LP3      LDA   54299     ;PRENDE UN VAL.
62          STA   $FB       ;CASUALE E LO
63          LDA   54299     ;NORMALIZZA
64          AND   #63       ;IN MODO CHE
65          ORA   #32       ;SIA COMPRESO
66          STA   $FC       ;TRA $2000
67          LDY   #0        ;E $3FFF
68          TYA             ;
69          STA   ($FB),Y   ;
70          LDA   161       ;CALCOLA IL
71          BEQ   LP3       ;TEMPO TRASCORSO
72          LDA   162       ;E SE E' FINITO
73          CMP   #224      ;ESCE
74          BCC   LP3       ;
75          LDA   #27       ;RIPRISTINA
76          STA   53265     ;I REGISTRI
77          LDA   #21       ;DELLA PAGINA
78          STA   53272     ;GRAFICA
79          LDA   #147      ;PULISCE LO
80          JSR   $FFD2     ;SCHERMO
81          RTS             ;ED ESCE
82 TABLE    DFB   $80,$C0,$F0,$10,$F0,$C0
83 TEMP     DFB   0
```

```
10 REM *------------------*
20 REM *   FRANTUMA VIDEO  *
30 REM *      C/64-128     *
40 REM * BY MICHELE MAGGI *
50 REM *------------------*
60 :
70 PRINTCHR$(147)"FRANTUMA VIDEO...."
100 FOR I=0 TO 156
110 READ A:CK=CK+A
115 PRINTCHR$(19)CHR$(17)"DATO N."I
120 POKE 16384+I,A
130 NEXT
140 :
150 IF CK=21364 THEN 170
160 PRINT"ERRORE NEI DATA...":END
170 SYS16384
180 :
1000 DATA 169,147,032,210,255,169,059
1001 DATA 141,017,208,169,024,141,024
1002 DATA 208,234,169,255,133,002,230
1003 DATA 002,165,002,201,006,240,243
1004 DATA 166,002,189,150,064,141,156
1005 DATA 064,162,155,032,179,238,202
1006 DATA 208,250,173,018,208,201,255
1007 DATA 208,249,173,156,064,162,000
1008 DATA 157,000,004,157,000,005,157
1009 DATA 000,006,157,232,006,232,208
1010 DATA 241,165,197,201,064,208,003
1011 DATA 076,020,064,234,169,000,141
1012 DATA 032,208,141,014,212,169,151
1013 DATA 141,015,212,169,129,141,018
1014 DATA 212,169,000,133,161,133,162
1015 DATA 173,027,212,133,251,173,027
1016 DATA 212,041,063,009,032,133,252
1017 DATA 160,000,152,145,251,165,161
1018 DATA 240,233,165,162,201,224,144
1019 DATA 227,169,027,141,017,208,169
1020 DATA 021,141,024,208,169,147,032
1021 DATA 210,255,096,176,192,240,016
1022 DATA 240,192,000
```

# Programma grafico universale

*E' possibile scrivere un programma che sia rigorosamente (o quasi) identico per C/16, C/128, Gw-Basic e Amiga? Noi abbiamo tentato...*

di Roberto Ferro

La rappresentazione grafica tridimensionalle di funzioni trigonometriche è una delle più affascinanti applicazioni che si possano realizzare con l'ausilio di un computer.

Data la grande facilità con cui gli elaboratori possono manipolare velocemente dati e calcoli, è stato possibile realizzare programmi che, dopo aver calcolato le coordinate dei punti di una funzione, la rappresentassero graficamente.

Questo è proprio ciò che compie il listato di queste pagine che rappresenta la funzione mediante un grafico tridimensionale (provare per credere!).

Il programma, grazie alla sua brevità e al raro riferimento ad istruzioni Basic specifiche solo di alcuni computer, può girare facilmente su qualsiasi modello ricorrendo, a seconda dei casi, a minime modifiche.

In questa occasione illustreremo gli adattamenti per C/16, Plus4, C/128, Ms-Dos e, udite udite, Amiga!

## Il programma

Come già detto, il listato è molto breve ed è questa caratteristica che lo rende estremamente versatile: in esso è contenuto solo il nucleo base per calcolare la funzione e rappresentarla.

Nulla vieta di arricchirlo aggiungendo altre opzioni quali la stampa o il salvataggio del grafico, a seconda delle possibilità offerte dal computer in uso.

Chi possiede un Amiga potrebbe, ad esempio, sfruttare i colori e l'alta risoluzione che questo magnifico computer offre.

Ai possessori di C/16 tutto ciò non sarà consentito: il listato va digitato senza errori e senza REM e non apportando alcuna modifica (pena il tradizionale "Out of memory error"): il tentativo di aggiungere linee di programma, come quello inserire spazi fra i comandi, potrebbe portare al completo esaurimento della memoria. Ciò non è dovuto tanto alla lunghezza del programma, quanto alla variabile dimensionata GR% che, immagazinando i punti del grafico, richiede molta memoria.

## Modificare la funzione

Chi desiderasse avere la rappresentazione di altre funzioni non dovrà far altro che modificare la linea 100. Il grafico prodotto dal programma senza modifiche rappresenta la funzione Y=SEN(X)/X.

## Adattamenti

Il listato pubblicato gira tranquillamente sul C/16 e sul C/128; chi ha fretta può attivare, a seconda del modello, la Poke (C/16) oppure i comandi Fast e Slow (C/128). Questi si ren-

dono necessari per velocizzare le lunghe procedure di calcolo.

Per l'Amiga, oppure l'Ms-Dos, gli adattamenti da apportare sono in numero maggiore ma, comunque, di lieve portata. I comandi per accelerare i calcoli (Fast, Poke) non sono nè disponibili nè necessari su Amiga; in AmigaBasic, infatti, i calcoli vengono effettuati in soli 12 secondi contro i 90 secondi del C/16 (e poco meno del C/128).

Le altre linee vanno modificate come segue:

*10 CLS(:SCREEN 2 - REM solo per gli Ms-Dos)*
*20 O=640:V=200:AL=1*
*30 IX =12:IY =3*
*220 LINE (XN,YN)-(XO,YO)*
*230 LINE (XN,YN)-(XO,YO)*

E' ovvio che il programma, così come è pubblicato, non sfrutta a fondo nè appropriatamente le capacità dei computer citati (soprattutto di Amiga, che non richiede la numerazione delle linee, ma la accetta egualmente) e, come già precisato, può essere arricchito e migliorato. Per chi volesse utilizzare altri interpreti che posizionano le origini degli assi in modo diverso, altre modifiche sono richieste anche alle linee 20 e 30.

```
1 REM PROGRAMMA GRAFICO, VERSIONE (QUASI) UNIVERSALE
2 REM C/16, C/128, PLUS/4, AMIGA, GW-BASIC
3 REM BY ROBERTO FERRO - TORINO
4 :
5 COLOR4,1:COLOR0,1:COLOR1,2:REM SOLO PER C/16 C/128
10 SCNCLR:GRAPHIC1,1
20 O=320:V=200:AL=1
30 IX=6:IY=3
40 F=INT(O/IX/2):G=INT(V/IY/3)
50 REM FAST (C/128)
55 REM POKE 65286,11 (C/16)
60 DIMGR%(F,G):IN=25
70 FORA=-G/2TOG/2
80 FORB=-F/2TOF/2
90 X=A*20/F:Y=B*20/G
100 Z=INT(SIN(SQR(X*X+Y*Y+.01))/(SQR(X*X+Y*Y+.01))*75+15*1.3)/200
110 GR%(B+F/2,A+G/2)=Z*AL*V
120 NEXTB,A
130 REM SLOW (C/128)
135 REM POKE 65286,27 (C/16)
140 FORY=1TOG
150 XB=IX*Y
160 YB=V/2+Y*IY+IN*AL
170 XO=XB+IX
180 YO=YB-IY-GR%(1,Y)
190 FORX=1TOF
200 XN=XB+X*IX
210 YN=YB-X*IY-GR%(X,Y)
220 DRAW1,XN,YNTOXO,YO
230 XO=XN:YO=YN:NEXTX,Y
240 FORY=1TOF
250 XB=IX*Y+G*IX
260 YB=V/2-Y*IY+G*IY+IN*AL
270 YO=YB-IY-GR%(Y,G-1)
280 XO=XB-IX
290 FORX=0TOG-1
300 XN=XB-X*IX
310 YN=YB-X*IY-GR%(Y,G-X)
320 DRAW1,XN,YNTOXO,YO
330 XO=XN:YO=YN:NEXTX,Y
340 END
```

# Simulatore di Amiga Info.

*Questa volta l'enciclopedia di routine L.M. offre un aiuto agli smemorati e a chi della precisione ha fatto un ideale di vita*

a cura di Alessandro de Simone

## Comsave (25375/25534)

A volte capita che, esaminando vecchi dischetti o cassette, ritroviamo il nome di un programma che, ormai, svolge solo la funzione di fare distinguere al computer un listato da un altro, ma non quella di richiamare alla memoria il contenuto del file stesso o il suo funzionamento.

Il nome di un file è troppo corto (max. 16 caratteri) anche per contenere una breve descrizione del programma, ma con un po' di buona volontà, qualche byte rubato alla memoria, e qualcun altro ai dischetti, abbiamo creato questa piccola routine la cui funzione è di registrare, prima del file principale, un file della lunghezza di circa 250 byte che conterrà sei righe di descrizione che potrà essere riletta con COMLOAD, comando pubblicato in queste stesse pagine.

La sintassi di COMSAVE è la seguente:

*SYS (XXXX), filename$[,dev]*

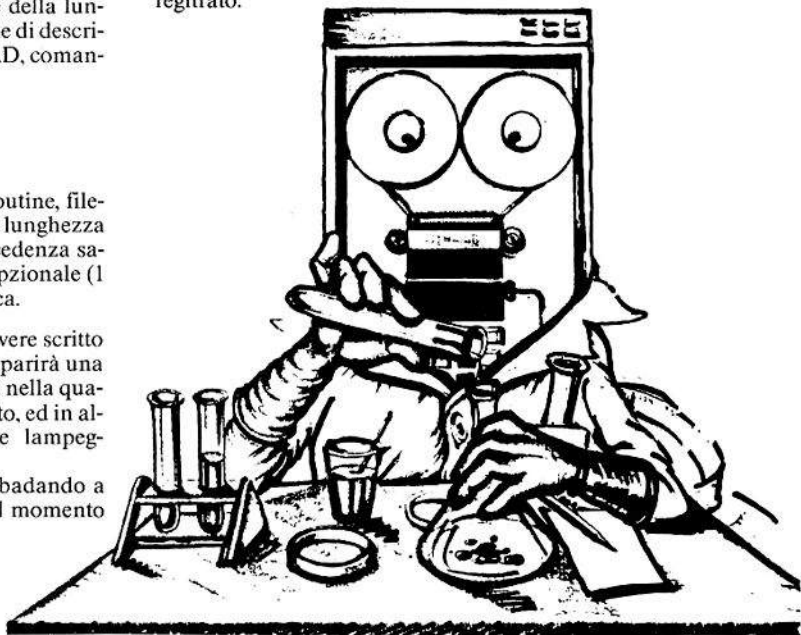
In cui XXXX è l'indirizzo di start della routine, filename$ è il nome del file (che potrà avere una lunghezza massima di quindici caratteri; quelli in eccedenza saranno troncati), mentre il parametro dev, opzionale (1 di default), indica il numero della periferica.

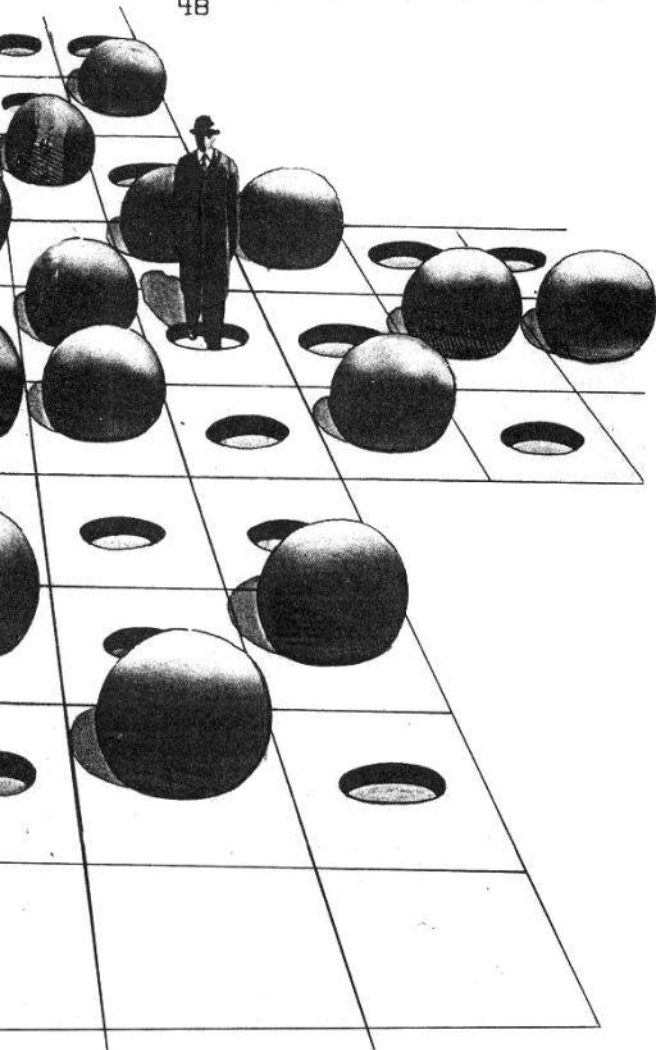
Una volta premuto il tasto Return, dopo avere scritto il comando, verrà cancellato lo schermo, apparirà una linea sulla settima riga, che delimiterà l'area nella quale si dovranno scrivere le parole di commento, ed in alto a sinistra sarà posizionato il cursore lampeggiante.

A questo punto si dovrà scrivere il testo badando a premere shift-return per andare a capo, dal momento che, come abbiamo detto, la pressione di return darà il via alla registrazione del file.

Se, durante la scrittura del commento, cambiamo idea e non vogliamo più registrarlo, basterà mettere in alto a sinistra nello schermo un asterisco (*); premendo return non sarà registrato alcun file, verrà cancellato lo schermo e si ritornerà al Basic.

Se, invece, diamo il via alla registrazione, vedremo apparire dapprima la scritta SAVING NOMEFILE. (con il punto finale) e poi SAVING NOMEFILE (senza il punto finale).

Ovviamente nomefile sarà il nome del file da noi specificato; il punto posto alla fine del primo nome indicherà che si tratta del commento al programma regitrato.

```
1000 PRINTCHR$(147)"COMSAVE"
1010 PRINT:PRINT"SYS XXXX,FILENA
     ME$[,DEVICE]
1020 PRINT:PRINT"FILENAME$=NOME
     FILE"
1030 PRINT"DEVICE=NUMERO PERIFER
     ICA"
1040 RETURN

1100 DATA 032,014,226,032,158,17
     3,032,163,182,201,015,144,0
     02
1110 DATA 169,015,168,132,002,17
     7,034,153,167,002,136,016,2
     48
```

```
1120 DATA 164,002,169,046,153,16
     7,002,162,001,032,121,000,2
     40
1130 DATA 006,032,253,174,032,15
     8,183,134,255,169,147,032,2
     10
1140 DATA 255,024,162,006,160,00
     0,032,240,255,160,040,169,1
     92
1150 DATA 032,210,255,136,208,25
     0,162,000,160,000,024,032,2
     40
1160 DATA 255,032,207,255,173,00
     0,004,201,042,208,005,169,1
     47
1170 DATA 076,210,255,024,162,00
     8,160,000,032,240,255,169,2
     55
1180 DATA 166,255,160,000,032,18
     6,255,166,002,232,138,162,1
     67
1190 DATA 160,002,032,189,255,16
     9,004,133,252,169,000,133,2
     51
1200 DATA 162,239,160,004,169,25
     1,032,216,255,169,255,166,2
     55
1210 DATA 160,000,032,186,255,16
     5,002,162,167,160,002,032,1
     89
1220 DATA 255,076,089,225,-1,209
     45
```

```
Programma: COMSAVE
Origin:$C000
By Fabio Sorgato

      JSR $E20E    ;cerca la virgola,
      JSR $AD9E    ;dopo prende una
      JSR $B6A3    ;stringa
      CMP #$0F     ;se la lunghezza di
      BCC *C00F    ;questa e' maggiore
      LDA #$0F     ;di 15, tronca la
*C00F TAY          ;parte eccedente.
      STY $02      ;salva la lunghezza
*C012 LDA ($22),Y  ;e poi mette la
      STA $02A7,Y  ;stringa a partire
      DEY          ;da $02A7
      BPL *C012    ;
```

```
        LDY $02      ;aggiunge un punto (.)
        LDA #$2E     ;alla fine dell nome
        STA $02A7,Y;del file
        LDX #$01     ;la periferica 1,
        JSR $0079    ;tape, e' di default;
        BEQ *C02E    ;se c'e' un'altro
                     ;carattere deve
        JSR $AEFD    ;essere una virgola,
        JSR $B79E    ;poi c'e' il numero
                     ;della periferica.
*C02E   STX $FF      ;Salva il numero della
                     ;periferica di default
                     ;o specificato
        LDA #$93     ;cancella lo schermo
        JSR $FFD2    ;
        CLC          ;posiziona il cursore
        LDX #$06     ;alla riga 6,
        LDY #$00     ;colonna 0;
        JSR $FFF0    ;
        LDY #$28     ;scrive 40 volte il
        LDA #$C0     ;carattere $C0();
*C041   JSR $FFD2    ;
        DEY          ;
        BNE *C041    ;
        LDX #$00     ;posiziona il cursore
        LDY #$00     ;alla riga 0,colonna 0
        CLC          ;
        JSR $FFF0    ;
        JSR $FFCF    ;accetta un testo da
                     ;tastiera aspettando
                     ;la pressione del
                     ;tasto RETURN.
        LDA $0400    ;Se il carattere
                     ;scritto in alto a
                     ;sinistra
        CMP #$2A     ;e' un asterisco (*)
        BNE *C05E    ;
        LDA #$93     ;cancella lo schermo
        JSR $FFD2    ;
        CLC          ;posiziona il cursore
        LDX #$06     ;alla riga 6,
        LDY #$00     ;colonna 0;
        JSR $FFF0    ;
        LDY #$28     ;scrive 40 volte il
        LDA #$C0     ;carattere $C0();
*C041   JSR $FFD2    ;
        DEY          ;
        BNE *C041    ;
        LDX #$00     ;posiziona il cursore
        LDY #$00     ;alla riga 0,colonna 0
        CLC          ;
        JSR $FFF0    ;
        JSR $FFCF    ;accetta un testo da
                     ;tastiera aspettando
                     ;la pressione del
                     ;tasto RETURN.
        LDA $0400    ;Se il carattere
                     ;scritto in alto a
                     ;sinistra
        CMP #$2A     ;e' un asterisco (*)
        BNE *C05E    ;
        LDA #$93     ;cancella lo schermo
        JMP $FFD2    ;ed esce
*C05E   CLC          ;altrimenti posiziona
        LDX #$08     ;il cursore alla riga
        LDY #$00     ;8,colonna 0
        JSR $FFF0    ;
        LDA #$FF     ;dispone il canale 255
        LDX $FF      ;con la periferica
                     ;specificata prima
        LDY #$00     ;e l'indirizzo
        JSR $FFBA    ;secondario a 0
        LDX $02      ;dispone il nome del
        INX          ;file compreso il
        TXA          ;punto aggiunto in coda
        LDX #$A7     ;a partire da $02A7
        LDY #$02     ;
        JSR $FFBD    ;
        LDA #$04     ;il file di commento
        STA $FC      ;iniziera' a $0400
        LDA #$00     ;
        STA $FB      ;
        LDX #$EF     ;e finira' a $04FB
        LDY #$04     ;
        LDA #$FB     ;
        JSR $FFD8    ;Registra il file
        LDA #$FF     ;mette gli stessi
        LDX $FF      ;parametri usati
        LDY #$00     ;prima
        JSR $FFBA    ;
        LDA $02      ;ma il nome del file
        LDX #$A7     ;non avra' il punto
        LDY #$02     ;finale.
        JSR $FFBD    ;
        JMP $E159    ;Registra ed esce
```

## Comload (25535/25753)

Dopo che il programma è stato registrato con il comando Comsave, se vorremo caricare solo il programma, o verificarlo, basterà usare la solita sintassi LOAD oppure VERIFY; se invece vorremo limitarci ad esaminare il commento, dovremo caricare COMLOAD in memoria e scrivere:

*SYS (XXXX), filename$[,device]*

in cui, come al solito, XXXX è l'indirizzo di partenza della routine comload, filename$ è il nome del file, device (1 di default) è il numero della periferica.

Verrà quindi cancellato lo schermo, apparirà la linea di delimitazione sulla settima riga e poi il testo del commento; comparirà, subito dopo, il messaggio LOAD (Y/N)?

Se vorremo caricarlo dovremo battere Y e premere return, altrimenti basta scrivere una N o un qualsiasi altro carattere; se avremo richiesto la lettura del file la scelta verrà segnalata dal solito messaggio LOADING FILENAME, altrimenti apparirà la scritta READY, e tutto ciò che era in memoria (escluso lo schermo, ovviamente), comprese le variabili, risulterà invariato.

Se il file non è presente sulla periferica, il commento non apparirà e, insistendo nel confermare la lettura del

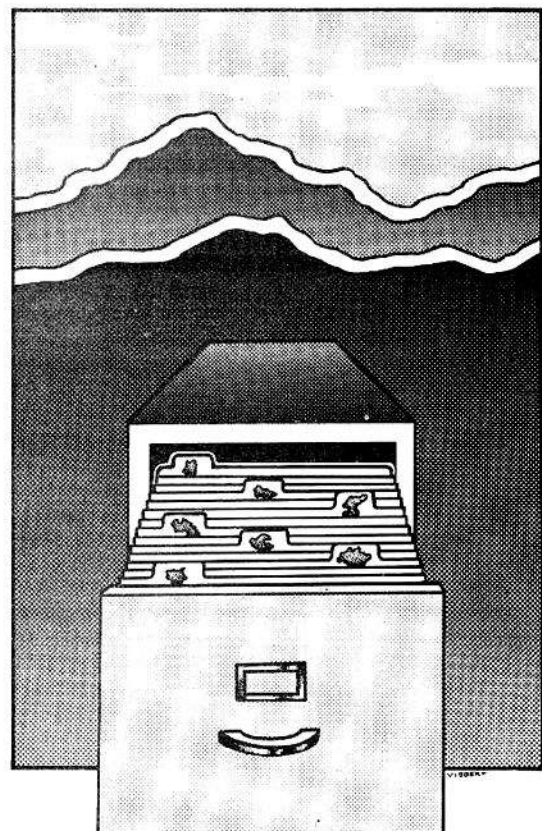
file, comparirà un meritato ?FILE NOT FOUND ERROR.

Osservando il disassemblato della routine si trova spesso il richiamo alla routine PLOT del kernal allocata da $FFF0; questa ha un duplice funzionamento: legge e fissa la posizione del cursore.

Per leggere la posizione del cursore si deve mettere il carry a 1 con SEC, eseguire un JSR alla routine e la coordinata X del cursore verrà riportata dal registro Y, mentre la Y dal registro X; se, invece, vogliamo fissare la posizione del cursore dovremo eseguire CLC, mettere X nel registro Y, Y nel registro X ed eseguire il salto alla routine.

Alla locazione C07B del disassemblato vediamo che la scritta LOAD del messaggio LOAD (Y/N)? viene letta dalla tabella dei comandi basic allocata da $A09E a $A128. Siccome tutti i comandi finiscono con l'ultima lettera SHIFT-ata viene cancellato il bit n.7 che indica, appunto, lo shift della lettera.

E' da notare che la seconda parte di tale messaggio è data da una serie di LDA immediati e di JSR alla CHROUT del kernal; avrebbe potuto essere racchiusa in una tabella, consentendo un notevole risparmio di memoria, ma le routine di questa enciclopedia hanno la caratteristica principale di essere totalmente rilocabili, mentre una tabella richiede un indirizzo assoluto che sarebbe stato necessario rilocare ogni volta con un procedimento che sarebbe sicuramente risultato più lungo.

(Le routine di questo numero sono di Fabio Sorgato)

```
1000 PRINTCHR$(147)"COMLOAD"
1001 PRINT:PRINT"SYS XXXX,FILENA
     ME$[,DEVICE]"
1002 PRINT:PRINT"FILENAME$=NOME
     FILE"
1003 PRINT"DEVICE=NUMERO PERIFER
     ICA"
1004 RETURN
1009 DATA 032,014,226,032,158,17
     3,032,163,182,201,014,144,0
     02
1010 DATA 169,014,168,132,002,17
     7,034,153,167,002,136,016,2
     48
1020 DATA 164,002,169,046,153,16
     7,002,162,001,032,121,000,2
     40
1030 DATA 006,032,253,174,032,15
     8,183,134,255,169,147,032,2
     10
1040 DATA 255,024,162,006,160,00
     0,032,240,255,160,040,169,1
     92
1050 DATA 032,210,255,136,208,25
     0,165,045,133,251,165,046,1
     33
1060 DATA 252,162,007,160,000,02
     4,032,240,255,169,255,166,2
     55
1070 DATA 160,000,032,186,255,16
     6,002,232,138,162,167,160,0
     02
1080 DATA 032,189,255,169,000,16
     2,000,160,004,032,213,255,1
     62
1090 DATA 000,169,013,032,210,25
     5,189,233,160,041,127,032,2
     10
1100 DATA 255,232,224,004,208,24
     3,169,032,032,210,255,169,0
     40
1110 DATA 032,210,255,169,089,03
     2,210,255,169,047,032,210,2
     55
1120 DATA 169,078,032,210,255,16
     9,041,032,210,255,169,063,0
     32
1130 DATA 210,255,169,032,032,21
     0,255,032,207,255,201,089,2
     40
```

```
1140 DATA 013,165,251,133,045,16
     5,252,133,046,169,013,076,2
     10
1150 DATA 255,169,255,166,255,16
     0,000,032,186,255,165,002,1
     62
1160 DATA 167,160,002,032,189,25
     5,169,000,076,113,225,-1,29
     563
1170 END
```

```
Programma: COMLOAD
Origin:$C000
By Fabio Sorgato

      JSR $E20E   ;cerca la virgola,
      JSR $AD9E   ;dopo prende una
      JSR $B6A3   ;stringa
      CMP #$0F    ;se la lunghezza di
      BCC *C00F   ;questa e' maggiore
      LDA #$0F    ;di 15, tronca la
*C00F TAY         ;parte eccedente.
      STY $02     ;salva la lunghezza
*C012 LDA ($22),Y ;e poi mette la
      STA $02A7,Y ;stringa a partire
      DEY         ;da $02A7
      BPL *C012   ;
      LDY $02     ;aggiunge un punto'(.)
      LDA #$2E    ;alla fine dell nome
      STA $02A7,Y;del file
      LDX #$01    ;la periferica 1,
      JSR $0079   ;tape, e' di default;
      BEQ *C02E   ;se c'e' un'altro
                  ;carattere deve
      JSR $AEFD   ;essere una virgola,
      JSR $B79E   ;poi c'e' il numero
                  ;della periferica.
*C02E STX $FF     ;Salva il numero della
                  ;periferica di default
                  ;o specificato
      LDA #$93    ;cancella lo schermo
      JSR $FFD2   ;
      CLC         ;posiziona il cursore
      LDX #$06    ;alla riga 6,
      LDY #$00    ;colonna 0;
      JSR $FFF0   ;
      LDY #$28    ;scrive 40 volte il
      LDA #$C0    ;carattere $C0();
*C041 JSR $FFD2   ;
      DEY         ;
      BNE *C041   ;
      LDA $2D     ;salva gli indirizzi
      STA $FB     ;della fine del
      LDA $2E     ;programma attualmente
      STA $FC     ;in memoria
      LDX #$07    ;e posiziona il
      LDY #$00    ;cursore a Y=7,X=0
      CLC         ;
      JSR $FFF0   ;
      LDA #$FF    ;dispone il canale 255
      LDX $FF     ;con la periferica
                  ;scelta
      LDY #$00    ;e il secondario 0
      JSR $FFBA   ;
      LDX $02     ;il nome del file
      INX         ;avra' un punto (.)
      TXA         ;aggiunto in coda
      LDX #$A7    ;
      LDY #$02    ;
      JSR $FFBD   ;
      LDA #$00   ;dispone il LOAD/VERIFY
      LDX #$00    ;flag a load e
      LDY #$04    ;legge il file a
      JSR $FFD5   ;partire da $0400
      LDX #$00    ;
      LDA #$0D    ;scrive un CR
      JSR $FFD2   ;
*C07B LDA $BDE9,X ;e la scritta LOAD
                  ;prendendola dalla
                  ;tabella dei comandi
      AND #$7F    ;e scartando il bit 7
                  ;acceso nell'ultimo
                  ;carattere
      JSR $FFD2   ;
      INX         ;
      CPX #$04    ;
      BNE *C07B   ;
      LDA #$20    ;scrive uno spazio
      JSR $FFD2   ;
      LDA #$28    ;scrive (
      JSR $FFD2   ;
      LDA #$59    ;scrive Y
      JSR $FFD2   ;
      LDA #$2F    ;scrive /
      JSR $FFD2   ;
      LDA #$4E    ;scrive N
      JSR $FFD2   ;
      LDA #$29    ;scrive )
      JSR $FFD2   ;
      LDA #$3F    ;scrive ?
      JSR $FFD2   ;
      LDA #$20    ;scrive uno spazio
      JSR $FFD2   ;
      JSR $FFCF   ;attende un carattere
                  ;da tastiera
      CMP #$59    ;se non e' Y
      BEQ *C0C4   ;
      LDA $FB     ;ripristina la
      STA $2D     ;fine del programma
      LDA $FC     ;in memoria
      STA $2E     ;
      LDA #$0D    ;scrive un CR
      JMP $FFD2   ;ed esce
*C0C4 LDA #$FF    ;usa sempre il canale
      LDX $FF     ;255 con la periferica
      LDY #$00    ;specificata e il SA=0
      JSR $FFBA   ;
      LDA $02     ;il nome del file e'
      LDX #$A7    ;senza il punto finale
      LDY #$02    ;
      JSR $FFBD   ;
      LDA #$00    ;load/verify flag=load
      JMP $E171   ;legge il file ed esce
```

Considerando che i numeri 1, 2 e 7 sono esauriti, vogliate inviarmi i numeri arretrati .......... al prezzo di L. 5.000 cadauno per richieste fino a 4 numeri, o di L. 4.000 cadauno per richieste oltre i 4 numeri arretrati, e perciò per un totale di L.............. Sono a conoscenza che i fascioli suddetti non saranno inviati in contrassegno e, pertanto, ho provveduto oggi stesso a versare il canone di L.................. a mezzo c/c postale n. 37952207 intestato a: Systems Editoriale - V.le Famagosta, 75 - 20142 Milano

**GIUDIZIO SUI PROGRAMMI DI QUESTO NUMERO**

Ho assegnato un voto da 0 a 10 ai programmi che indico di seguito:

A/ .......................................... Voto ..................

B/ .......................................... Voto ..................

C/ .......................................... Voto ..................

D/ .......................................... Voto ..................

**PICCOLI ANNUNCI** ..........................................

..........................................

..........................................

..........................................

..........................................

..........................................

**CERCO/OFFRO CONSULENZA** ..........................................

..........................................

..........................................

..........................................

..........................................

..........................................

## INVIARE IN BUSTA CHIUSA E AFFRANCANDO SECONDO LE TARIFFE VIGENTI A:

## COMMODORE COMPUTER CLUB

## V.le Famagosta, 75
## 20142 Milano

**INVIARE TUTTA LA PAGINA ANCHE SE SI UTILIZZA UNA SOLA SCHEDA**

Nome .................................... Cognome ....................................

Via .................................... N° ............ CAP. ............ Città ....................

Telefono .................................... Orario ....................................

# Quale fascicolo manca alla tua enciclopedia Commodore?

*Per ordinare i fascicoli mancanti alla tua collezione di Commodore Computer Club utilizza l'apposita scheda in fondo alla rivista.*

# Il peso della velocità (e altre storie)

*Alcuni ragionamenti uniti a pochi e semplici listati, possono evidenziare le caratteristiche peculiari del nostro computer. Questo inserto rappresenta anche un invito a soffermarsi sulle potenzialità di un calcolatore che maggiormente dovrebbero interessare l'utente finale.*

**di Alessandro de Simone**

## Il bello, il brutto, l'inutile

Troppo spesso, nella presentazione di una nuova apparecchiatura, i vari autori si dilungano su argomenti che ritengono di prevalente importanza.

Nella valutazione di una "macchina", che la moderna tecnologia mette a disposizione, troppo spesso si trascurano molte delle caratteristiche fondamentali che costituiscono, tuttavia, le principali motivazioni dell'acquisto.

Per esempio, volendo acquistare un moderno apparecchio TV, dovrebbe esser necessario considerare soprattutto la distanza alla quale, di solito, si assisterà agli spettacoli televisivi.

La scelta di un apparecchio grande, da sistemare in una stanza di modeste dimensioni, è da scartare a priori: tutti sanno che un grande schermo, osservato da distanze relativamente ridotte, non potrà nascondere le linee caratteristiche della scansione video, provocando, di conseguenza, una visione fastidiosa.

Un apparecchio piccolo, al contrario, non dovrebbe esser sistemato in un ambiente di dimensioni notevoli, pena l'impossibilità di distinguere i particolari più piccoli delle immagini teletrasmesse.

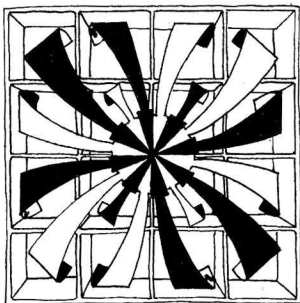
Analogamente, bazzicando nei negozi di apparecchiature Hi-Fi, si incontrano persone, che si autodefiniscono "intenditori", che confrontano attentamente i valori dichiarati dalle case costruttrici sulle distorsioni, sui decibel e sulla separazione dei canali; questi individui consultano tabelle, si scambiano pareri in cui sono chiamate in causa unità di misura e metodologie di misurazione straordinariamente sofisticate, più adatte alla ricerca spaziale che non ad un pacifico ascolto di musica.

Queste stesse persone, però, non dedicano la minima attenzione alle caratteristiche acustiche del locale in cui verrà sistemato l'impianto Hi-Fi.

La colpa di questo strano comportamento, a dire la verità, è imputabile anche alle riviste specializzate del settore, che troppo spesso trascurano di riportare considerazioni sulla opportunità di mettere in evidenza il ruolo determinante che assume l'ambiente acustico.

Molto più spesso di quanto non si creda, infatti, l'arredamen-

to del locale limita fortemente la correttezza della riproduzione sonora, vuoi a causa di pareti riflettenti, vuoi per l'eccessivo assorbimento di alcune frequenze rispetto ad altre, e così via.

Pochi sanno che una cuffia acustica di basso prezzo riproduce la musica in modo decisamente più fedele e "coinvolgente" di un paio di casse acustiche di notevole prezzo sistemate, però, in un ambiente acusticamente trascurato.

## C'era una volta il computer

Un moderno calcolatore non sfugge all'analisi, e alle critiche, cui abbiamo prima accennato.

La mentalità con cui viene criticamente valutato oggi un nuovo modello rivela, però, un tipico atteggiamento che si poteva giustificare solo durante la fase "arcaica" della storia del

computer; ci riferiamo, ovviamente, al periodo in cui l'attenzione degli utenti era attratta prevalentemente da considerazioni dettate da ben altre premesse.

Ogni epoca ha i suoi problemi e quando questi vengono brillantemente risolti, si tende, giustamente, a porre in rilievo i risultati raggiunti.

La quantità di memoria Ram disponibile, nel corso degli anni '70, era un argomento di fondamentale importanza dal momento che, soprattutto per procedure contabili, la gestione di un archivio veniva enormemente facilitata. La velocità delle memorie magnetiche, inoltre, non consentiva di supplire, in modo soddisfacente, ricorrendo alla tecnica della memoria virtuale in cui, come è noto, il disco viene considerato come una... protesi cerebrale del banco di memoria Ram: la velocità di trasferimento dei dati, infatti, non consentiva elaborazioni agevoli nè sotto il profilo della praticità nè della velocità.

Calcolatori che disponessero di "grandi" quantità di memoria Ram, quindi, vennero accolti con notevole soddisfazione dagli operatori del settore che definivano gigantesca (non dimentichiamo l'epoca alla quale ci riferiamo!) la possibilità di "espandere il sistema da un minimo di 8 fino a ben 32 K Ram", come ci teneva a mettere in evidenza la pubblicità di quel periodo.

Il progresso tecnologico ha apportato combiamenti così rapidi, sui quali non ci soffermiamo, che gli stessi addetti ai lavori non sembrano essersi abituati alle novità.

Lo dimostra l'insistenza nel considerare determinante la quantità di memoria Ram (dal prezzo odierno, peraltro, bassissimo), oppure la disponibilità di caratteristiche che, al giorno d'oggi, non rivestono più l'importanza di un tempo.

Facciamo un esempio: la pubblicità di alcuni modelli economici MSX pone l'accento sul fatto che sono disponibili 256 K Ram. Pur se, a onor del vero, viene specificato che 128 K sono destinati esclusivamente alla gestione della grafica, a nessuno viene in mente di valutare, in termini di utilizzo reale, una disponibilità talmente vasta.

Dovrebbe esser noto, ma non ci stancheremo di ripeterlo, che un computer non serve assolutamente a nulla se privato del software per farlo funzionare; non si può pretendere, nell'ambito di una tendenza alla massificazione dell'informatica, che ognuno scriva i programmi che intende utilizzare.

Eppure, paradossalmente, molte fabbriche di computer insi-

stono, nei loro "comunicati commerciali" a mettere in evidenza le caratteristiche hardware piuttosto che quelle software; solo una multinazionale, la IBM, ha basato una sua campagna promozionale, molto intelligente, sulla disponibilità del software: nelle inserzioni, infatti, compariva l'ormai nota immagine di Charlot circondato da una massa notevole di pacchetti software relativi ad ogni genere di applicazioni.

Ma, si sa, non tutti la pensano allo stesso modo...

## Finchè c'è Ram c'è speranza

Continuiamo ad occuparci di Ram. Lavorando con un word processor si sente spesso l'esigenza di poter scrivere testi di ampio respiro, senza esser costretti a dividerli in più parti.

In casi simili è indispensabile poter disporre di una quantità elevata di Ram e questo fatto è ben noto a chi possiede un Plus 4 (che consente di stendere documenti di sole 100 righe) a dispetto dei 64 K Ram montati sulla macchina.

Easy Script, il w/p di cui ci occupiamo spesso, consente di scrivere testi lunghi fino a 764 righe di 40 caratteri ciascuna. Il w/p Superscript, che gira sul C/128, permette testi di 764 righe ma "larghi" fino ad 80 caratteri (per non parlare di una parte "secondaria", utile per la manipolazione di porzioni ridotte di testo).

Sembrerebbe, quindi, che con un modesto C/64 (oppure un C/128) sia aggirabile la limitazione della lunghezza di un testo, ma un utilizzatore, che abbia un minimo di buon senso, non si permetterà mai di scrivere, di seguito, un testo così lungo; in agguato c'è sempre l'imprevisto che, nel caso specifico, si chiama "mancanza di corrente".

Vi è mai capitata, nel bel mezzo di un lavoro impegnativo, l'interruzione, anche per la durata di un solo attimo, della fornitura di energia elettrica? Naturalmente questi fenomeni capitano non all'inizio del lavoro ma, diabolicamente, quando la fatica è quasi alla fine, con le conseguenze che è facile immaginare.

Chiunque sia stato scottato da un'esperienza come quella descritta, e che sia ancora sano di mente, ha imparato che nel corso di un lavoro al computer (qualunque esso sia) è buona norma registrarne le fasi intermedie in modo che, in caso di infortunio, non sia necessario ricominciare tutto daccapo, ma solo dalla fase relativa all'ultimo salvataggio effettuato.

In pratica non serve avere a disposizione una grande quantità di Ram se il software non prevede un'elaborazione "sicura", al riparo da inconvenienti di vario genere che dovessero capitare.

In tale ottica si inserisce perfettamente la nostra insistenza nel consigliare la disponibilità di un disk drive; chi adopera il solo registratore a cassette rischia spesso, a causa delle lentezza tipica di quest'ultima periferica, di rinunciare alle operazioni intermedie di salvataggio e, in ogni caso, si è costretti ad una minore velocità generale di gestione dei dati, in aperta contraddizione con la stessa "filosofia" informatica, che dovrebbe essere caratterizzata, principalmente, da elevate velocità operative.

A maggior ragione, non ha senso utilizzare computer più potenti (come il C/128) privi di un disk drive, oppure computer grafici di un certo pregio che ricorrono a televisori domestici, capaci, questi ultimi, solo di mortificare le notevoli potenzialità del calcolatore.

Se cerchiamo, con altre parole, di riassumere quanto detto, possiamo affermare che l'attribuire importanza ad alcune caratteristiche, trascurandone altre, tende a viziare il giudizo che si può dare di una macchina.

Ad esempio, il problema della sicurezza, e di evitare la perdita accidentale dei dati, è affrontato, e risolto, in vari modi. Per ciò che riguarda i testi elaborati con un w/p, molti programmi di trattamento dei testi provvedono automaticamente a registrare porzioni di testo, a mano a mano che l'operatore li compone. Se dovesse mancare l'energia elettrica, quindi, sarà possibile rintracciare, su disco, il file più recente, ricaricarlo e proseguire il lavoro. Tale modo di operare, ovviamente, presuppone una velocità elevata di scambio dei dati tra computer e periferica (il drive, in questo caso) che garantisca una trasmissione dati veloce o, addirittura, "trasparente" intendendo, con tale termine, la possibilità che l'operazione avvenga senza che l'operatore se ne accorga o che intervenga direttamente per effettuarla.

## Verso un'architettura armonica

Un moderno calcolatore non può esser progettato senza tener conto dei numerosi fattori che ne determinano la reale versatilità di utilizzo.

Abbiamo visto, in precedenza, che non ha senso contare il numero dei byte disponibili se, all'atto pratico, non c'è modo di utilizzarli tutti in modo "sicuro". Anche in questo caso ricorreremo ad un esempio che meglio chiarirà quanto intendiamo affermare.

Nella produzione dei calcolatori domestici dei primi anni '80 si preferì ricorrere ad una formula che, in effetti, si dimostrava, per quei tempi, ottimale. Ogni computer disponeva, su Rom, non solo di un sistema operativo ma anche, e soprattutto, di un linguaggio che, di solito, era il Basic.

Tale soluzione era il risultato di una serie di considerazioni che mettevano al primo posto la semplicità operativa. L'utilizzatore non doveva far altro che accendere il computer per trovarsi in un "ambiente" di facile gestione: chi aveva intenzione di caricare un programma da periferica, come pure chi intendeva scriverne uno, era egualmente facilitato nel compito. La cultura informatica dell'epoca, d'altra parte, non permetteva di allontanarsi dal Basic che, ancora oggi, rappresenta il linguaggio più facilmente assimilabile dalla maggior parte dei principianti.

La decisione di dedicare una parte dell'area indirizzabile a qualche Rom che contenesse Sistema e Linguaggio era quindi la migliore (e, diciamo la verità, la più economica e la più "vendibile"); le difficoltà derivanti dalla inevitabile rigidità di un sistema così concepito si avvertiva a livelli più profondi.

Lavorando in linguaggio macchina, infatti, i programmatori delle software house, ma anche gli hobbysti evoluti, non potevano fare a meno di considerare, lo spazio occupato dalle Rom, una occupazione inutile di memoria e l'impossibilità di caricare, nella stessa area, routine che già possedevano e che funzionavano perfettamente in altri ambienti gestiti dallo stesso microprocessore.

In altre parole, i programmatori abituati a lavorare con grossi calcolatori basati sullo Z-80, uno dei migliori microprocessori disponibili ad otto bit, si ritrovavano un'area più angusta, a causa delle Rom presenti nei piccoli calcolatori domestici, e non riuscivano a realizzare programmi che, in altre condizioni, avrebbero sicuramente potuto scrivere.

## Rom mia, per piccina che tu sia...

Nei sistemi più grossi, di solito, il microprocessore ha a sua disposizione l'intera area indirizzabile riempita quasi esclusivamente da Ram.

Oltre ai chip per le operazioni di ingresso e uscita, infatti, è presente, nei calcolatori più "aperti", una Rom di minima dimensione (a volte solo un paio di K Ram) che ha il compito limitato di effettuare il Bootstrap (=caricamento).

Quando si accende uno di questi calcolatori compare, sullo schermo, la richiesta di inserire il disco di sistema nel drive. Se, a questo punto delle operazioni, si tenta di premere i tasti della tastiera non si ottiene alcun risultato. Alcuni sistemi riconoscono anche la pressione di un tasto qualsiasi, operazione che comunica al sistema l'avvenuto inserimento del disco; altri sistemi, come l'Amiga, escludono del tutto la tastiera perchè riescono a "capire" se un disco è stato inserito nel drive e se, naturalmente, è proprio il disco richiesto.

Un computer, basato su un'architettura del tipo descritto, è considerato aperto, nel senso che, conoscendo il modo di operare della Rom di bootstrap, è possibile sostituire, in qualsiasi momento, sistema operativo o linguaggio, consentendo, quindi, un'ampia possibilità operativa sia all'utilizzatore che, soprattutto, al programmatore.

Al microprocessore, come è noto, non interessa sapere se il programma da eseguire è riportato su supporto Rom oppure Ram; ciò che importa è che il programma abbia una sua coerenza e che conduca ai risultati voluti. Caricando da disco il programma, questo potrà essere allocato esattamente nelle locazioni previste e se, un domani, ci si dovesse accorgere che una certa routine può essere sostituita da un'altra più efficiente, sarà sufficiente alterarla e sostituirla con quella vecchia nel modo più semplice possibile: registrando la nuova versione su disco e abbandonando la vecchia procedura.

Un tale modo di operare ha consentito di realizzare più versioni successive di Ms-Dos, sempre migliori ripetto alle precedenti e che girano, naturalmente, sugli stessi calcolatori prodotti anni addietro; nel caso dell'Amiga si è giunti alla versione 1.2 che, con una semplice sostituzione dei dischetti, permette potenzialità più raffinate della precedente versione 1.1; e siamo solo all'inizio.

L'unica seccatura di un sistema operativo "soft" (cioè su disco) è rappresentata dal fatto che non è possibile iniziare a lavorare immediatamente non appena si accende il calcolatore ma, a seconda dei casi, è necessario inserire uno o più dischi in successione.
Per ciò che riguarda i programmi professionali, tuttavia, un sistema aperto è quanto di meglio esista oggi sul mercato perchè permette ai programmatori una libertà inconcepibile in sistemi basati su Rom che occupino larghe aree di memoria e che vengono definiti, di conseguenza, "chiusi".

## La guerra dei bit

Un'altra attenzione particolare è spesso rivolta al numero dei bit del microprocessore su cui è basato un sistema computerizzato; ed anche in questo campo le perplessità nascono da considerazioni acritiche portate avanti con notevole disinvoltura.
Lungi dal voler rivelare verità inconfutabili, ci limiteremo, in questa sede, a seguire ragionamenti di vario tipo che abbiano lo scopo di affrontare il problema da un punto di vista pratico.
Troppo spesso si è abituati a considerare "migliori" di altri, alcuni oggetti che dispongono di una quantità maggiore di "cose". Da questo punto di vista un microprocessore a 16 bit sembrerebbe avvantaggiato, in un confronto di questo tipo, rispetto ad un micro di appena otto bit.
Pur se, in linea di massima, si può esser d'accordo con affermazioni del genere citato, tuttavia sarebbe necessario considerare un calcolatore nel suo insieme e non limitandosi a potenzialità non meglio dimostrate.
La storia commerciale dell'informatica annovera, purtroppo, numerosi esempi negativi di computer che, basati su microprocessori di tutto rispetto, hanno fatto una brutta fine. Ci riferiamo ad "Ada", che rappresentò una spina nel fianco della prestigiosa Apple, al TI-99 della Texas (che ancora si lecca le ferite), alla versione "strana" del 68000 montato sull'ancora più strano QL Sinclair (che ha causato il tracollo finanziario di Sir Clive).
Paradossalmente il micro ad otto bit 6502, che alla sua nascita fu considerato senza futuro, ha fatto la fortuna della Apple, della Commodore e di altre Case che, in seguito, lo vollero adottare.

Il C/64, anzi, dimostrò che un computer ad otto bit, e per giunta realizzato seguendo lo schema di un'architettura chiusa, può incontrare più favori di uno Z-80 che sovrintende ad una architettura molto più aperta (e chi ha orecchie per intendere, intenda...).

## Off Limits

Perchè mai, dunque, si tende ad abbandonare le architetture degli otto bit, sperimentate e sicure, a favore di sistemi basati sui 16 bit?
La risposta è piuttosto semplice; ogni macchina è figlia dell'epoca in cui nasce ed è destinata a seguire l'evoluzione dei gusti. Quando il C/64 fu presentato, rappresentava quanto di meglio si potesse pretendere da un computer di una certa fascia di prezzo.
Con il passare del tempo, però, l'utente finale si è abituato alle potenzialità del piccolo computer e le sue esigenze, quindi, sono cresciute di conseguenza. A nulla sono valsi gli interventi sulle periferiche, che dotate di speed dos, e analoghi velocizzatori, hanno consentito di raggiungere velocità di trasferimento paragonabili a quelle offerte da computer della categoria di un Ms-Dos; particolari gestioni grafiche e sonore richiedono, inevitabilmente, un mare di Ram ed una velocità operativa che sia all'altezza della situazione.

Una pagina video di normali dimensioni (80 colonne x 25 righe) richiede 2000 byte se adoperata in modo testo e ben 16000 se in alta risoluzione. Un microprocessore in grado di gestire 64 K non può assolutamente essere in grado di manipolare programmi grafici di una certa complessità che richiedano, ad esempio, animazioni, modifiche del colore, memorizzazione e successivi richiami di pagine grafiche elaborate.

Pur se, a prezzo di particolari accorgimenti, è possibile gestire più banchi da 64 K Ram ciascuno, un microprocessore ad otto bit presenta pesanti limitazioni. La principale di queste è sicuramente rappresentata dalla frequenza massima alla quale può operare un microprocessore; non è sufficiente (magari lo fosse!) cambiare il quarzo per costringere il micro ad operare a velocità maggiori. Superando una determinata frequenza insorgono fenomeni negativi, quali autooscillazioni, perdita

di dati, mancata risposta dei chip di memoria, che impediscono, di fatto, l'elaborazione.
Un microprocessore a 16 bit, invece, è realizzato con le tecniche più moderne, derivanti proprio dalle esperienze acquisite operando nel mondo degli otto bit: sarebbe stato riduttivo, insomma, limitarsi, in un nuovo progetto, negli ambiti ristretti degli otto bit quando l'esperienza consentiva di concretizzare ben altre ambizioni.

## Più banchi, più onore

Vedremo ora di spiegare, seppur per sommi capi, in che modo un microprocessore ad otto bit può gestire una quantità di memoria maggiore di 64 K.
Prima di andare avanti è opportuno ricordare che l'indirizzo di un byte, appartenente alla memoria gestibile dal micro, deriva (di solito) dal calcolo che coinvolge due byte posti in successione tra loro (puntatori).
Premesso che un byte può contenere un valore intero variabile tra 0 e 255, se al primo di due byte successivi assegniamo il valore che rappresenta (tale e quale) e al secondo, invece, attribuiamo il significato di "pagina", l'indirizzo di un qualsiasi byte potrà esser individuato considerando la sua posizione all'interno di una ben precisa pagina.
Ad esempio, il byte individuato dai due valori successivi 3 e 10 (che rappresentano, nella nostra ipotesi, il "codice" di posizione relativo al byte considerato) sarà quello il cui indirizzo è 2568 (=10*256+3). Ecco perchè l'indirizzo più basso è zero (=0*0+255*0) ed il più alto 65535 (255+255*256) valore che, in gergo, corrisponde a 64K.
Bisogna sapere, a questo punto, che un chip di memoria (circuito integrato Ram oppure Rom, a seconda dei casi) possiede, tra gli altri, otto pin (piedini) per i dati (consideriamo sistemi ad otto bit) e sedici pin per gli indirizzi, oltre ai pin per l'alimentazione e ad altri pin.
In effetti non tutti i dispositivi di memoria possiedono i 16 pin degli indirizzi; questo perchè la gestione dell'indirizzamento può esser realizzata con altre tecniche che ricorrono, per selezionare il chip interessato, ad altri chip logici.
Ciò che importa sottolineare, invece, è che un chip di memoria dispone di un pin che si chiama "Enable" (=abilitazione). Questo particolare piedino, a seconda se posto a massa oppure

a +5 volt, abilita, o meno, lo stesso chip a ricevere, oppure fornire, dati.

Non è sufficiente, dunque, che il micro, per individuare un byte, selezioni un certo indirizzo che, fatti i debiti calcoli di decodifica, appartenga al chip interessato: se il pin Enable non è opportunamente attivato, il dato non potrà essere letto (o scritto) nel chip individuato.

E' come se voi saliste ad un certo piano di un condominio: potete entrare in un appartamento del piano solo se possedete la chiave, altrimenti... restate sul pianerottolo!

In sistemi semplici, come nel vecchio Vic 20, in cui non sono possibili equivoci, i pin Enable dell'intera memoria (Rom oppure Ram che sia) sono sempre attivati (non è proprio così, ma a livello generale il ragionamento può esser considerato valido); è come se doveste entrare ed uscire da appartamenti di un condominio in cui, a ciascun piano, esiste un solo alloggio la cui porta è prevista sempre aperta: non avrebbe senso infatti (ma solo in sistemi semplici), giunti al piano interessato, avere un ulteriore strumento di identificazione per entrare nell'alloggio (cioè la chiave = l'abilitazione del chip selezionato).

Possiamo, però, strutturare il nostro sistema in modo leggermente diverso; pur se disponiamo di un solo ascensore (leggi: microporocessore capace di gestire 256 pagine di 256 byte ciascuna), possiamo sistemare, per ciascuno dei 256 piani (:pagine), più appartamenti (:chip di memoria) che hanno quindi in comune lo stesso pianerottolo (:numero di pagina).

Al microprocessore sarà sufficiente, nelle fasi di gestione della memoria, comunicare, volta per volta, in quale appartamento entrare, subito dopo essersi posizionato sul pianerottolo voluto.

In questo modo sarà possibile, ad esempio, inserire due chip di memoria nella stessa "zona" nonostante abbiano gli stessi indirizzi. Tale tecnica è proprio quella utilizzata nel C/64 che dispone, sovrapposti tra loro, di alcuni chip Rom (linguaggio e sistema operativo) e di altrettanti chip Ram.

Quando si utilizza il Basic, il 6510, micro del C/64, rende attivi i soli chip Rom (che contengono linguaggio e sistema) e inattivi quelli Ram posti al di sotto. Lavorando in linguaggio macchina è tuttavia possibile rendere attiva la Ram posizionata "sotto" le Rom e, contemporaneamente, inibire queste ultime; non è possibile, infatti, attivare nello stesso momento due chip che abbiano lo stesso indirizzo.

E' intuitivo che sarà possibile non limitarsi a due soli chip allocati nello stesso indirizzo; lavorando con po' di fantasia (e con un po' di hardware aggiuntivo), si potranno "sovrapporre" un numero praticamente infinito di chip e selezionare, volta per volta, quello che più interessa.

Al limite è possibile "sovrapporre" una diecina di Rom contenenti, ciascuna, un programma professionale (word processor, spreadsheet, data base ed altri) che agisca, o meno, su una propria area Ram oppure su un'area comune realizzando koctail di gestione fantasiosi come si vuole.

Ed è proprio ciò che è stato fatto con il Plus 4 (più Rom professionali selezionabili una per volta) e con il C/128, che può gestire fino a 16 banchi di memoria da 64 K ciascuno.

Anche per il C/64 sono state proposte espansioni incredibili di Ram e di Rom, sempre sfruttando il semplice trucchetto di cui abbiamo parlato.

## Un otto volante

Dopo aver letto i paragrafi precedenti verrà spontaneo chiedersi perchè mai le fabbriche di computer non si limitino ad utilizzare micro ad otto bit che, in un modo o in un altro, gestiscano la quantità di memoria desiderata.

La risposta è piuttosto semplice: un micro ad otto bit, sottoposto a sforzi così grandi, deve pur presentare il conto di acrobazie tanto spericolate. C'è da rilevare, infatti, che per individuare un byte appartenente ad un chip che possiede il pin enable costantemente attivo, occorre un tempo decisamente breve; nel caso di più chip da selezionare, prima di accedervi, inevitabilmente il micro deve impiegare un certo periodo di tempo, necessario alla selezione del chip, la cui durata complessiva si fa rilevante, specialmente nel corso di elaborazioni lunghe e complesse.

Un micro ad otto bit, insomma, deve volare da un punto all'altro della memoria e, per giunta, selezionare uno o più banchi; se, poi, si tiene conto che per una corretta azione si rende anche necessario alterare puntatori, ricopiarne contenuti, risistemarli dopo aver effettuato il salto (senza trascurare eventuali compiti richiesti dall'interrupt), si può comprendere come il tempo trascorra con enorme facilità, annullando, in tutto o in parte, i vantaggi derivanti da una maggior disponibilità di memoria.

Per facilitare il compito, e limitare i "danni" provocati dai salti, sono state studiate versioni sempre più veloci e versatili di uno stesso microprocessore; tale sorte è stata riservata anche al 6502 (montato sulla serie Pet e sul Vic 20), trasformato dapprima in 6510 (Commodore 64), poi in 8501 (C/16 e Plus/4) ed infine in 8502 (C/128). Questi micro dispongono delle stesse istruzioni in linguaggio macchina (compatibilità a livello Assembly), si differenziano tra loro per una diversa architettura harware interna (e, come tale, "trasparente" per l'utente) ma, soprattutto, per una diversa gestione di alcune locazioni di memoria che comunicano al sistema quale banco selezionare in un determinato momento.

Sofisticare ulteriormente un microprocessore diventa, alla fine, antieconomico; pensate ad un'automobile come la Fiat 500: dopo aver potenziato il motore, insonorizzato l'interno, resi più comodi(!) i posti per i passeggeri, disposto le portiere controvento ed abbellito la carrozzeria migliorando il disegno dei paraurti, dei gruppi ottici e dei finestrini laterali, il passo successivo per apportare migliorie è rappresentato unicamente dalla produzione della... Panda! Con ciò vogliamo affermare che un prodotto, pur se sperimentato e migliorato nei minimi particolari, non può offrire prestazioni che superino un certo livello.

Allo stesso modo, nel campo dei computer, è molto meglio utilizzare un 16 bit invece che cercare, poco razionalmente, di migliorare un micro che ha già offerto, in pratica, tutto ciò che poteva offrire, tenendo conto dell'architettura dalla quale si era partiti.

## Meglio un uovo oggi...

Ma il discorso (poteva essere diversamente?) non è così semplice; anche rassegnandosi all'idea di cambiare micro, rimane il compito, decisamente impegnativo, di sceglierne uno tra quelli che il mercato offre. Come si può intuire, la scelta deve esser condizionata da tantissimi fattori: economia, disponibilità sul mercato, possibilità di disporre di versioni successive, studio della gestione di chip che il progresso tecnologico metterà tra breve a disposizione, legame ad una "famiglia" di computer piuttosto che ad un'altra, sperimentazione, affidabilità, eccetera.

Nella scelta di un micro, insomma, il successo di un computer viene legato indissolubilmente alle "cose" che questo micro potrà permettere di realizzare; e si rimane fuori, ovviamente, dalle "cose" che altri micro potrebbero, in un immediato futuro, consentire.

Del resto è possibile anche commissionare alla fabbrica dei micro una versione particolare, che assolva ad alcune funzioni di cui si avverta la carenza. Ma questi studi richiedono investimenti di denaro non indifferenti, che richiedono tempo ed il tempo, si sa, è denaro anch'esso...

La guerra dei 16 bit, che ci piaccia oppure no, è oggi combattuta prevalentemente da una rosa di candidati piuttosto ristretta: serie Intel 80X86 (8086, 80286, 80386) posta alla base dei sistemi Ms-Dos (IBM, AT, XT e compagnia); la serie Motorola 68000 (Amiga, Atari, Macintosh); la serie Zilog Z8000 (minicomputer di marche meno note ai nostri lettori).

Leggendo riviste americane di elettronica digitale, tuttavia, si scopre che il mercato è letteralmente invaso da una miriade di altri micro, ognuno diverso dall'altro, e dotato di caratteristiche di tutto rispetto; ma sono, questi, micro destinati ad applicazioni molto particolari, piuttosto specifiche (soprattutto per scopi militari e per telecomunicazioni), che svolgono in maniera egregia alcune funzioni, ma non saprebbero come fare per svolgerne altre di uso più "comune".

E non è finita: dopo aver scelto il micro è indispensabile scegliere il "contorno"; e qui la scelta è ancora più vasta, dal momento che chip particolari (grafica, suono, I/O) possono anche essere commissionati su misura se i modelli offerti dal mercato non soddisfano pienamente.

In ogni caso, divincolandosi nel marasma che sicuramente abbiamo provocato in chi ci ha seguito fin qui, possiamo ragionevolmente affermare diverse cose, alcune di stampo Pazzagliano, altre più credibili:

- E' meglio disporre di un sistema ad otto bit realizzato razionalmente che uno a 16 bit mal progettato.
- E' meglio un sistema ad otto bit veloce che uno ad otto bit lento.
- E' meglio un sistema aperto che uno chiuso.
- E' meglio un sistema "vecchio" e sperimentato basato su un micro ad otto bit che un sistema nuovo basato, anch'esso, su un otto bit: il progresso non consente nostalgie.

Pur se le "verità inconfutabili" appena predica... enunciate possono sembrare banali ed irrilevanti, cercheremo di dimostrare che alcune di esse si presentano tutti i giorni sotto i nostri occhi e, magari, non ce ne rendiamo conto.

## Io Chip, tu matto

Se pensiamo ad un linguaggio, il Basic, che conosciamo benissimo(!), forse non ci rendiamo conto che, in fin dei conti, non è altro che un gruppo di routine in linguaggio macchina concatenate tra loro.

Molti pensano, purtroppo, che vi sia una differenza netta tra istruzioni Basic, istruzioni in linguaggio macchina, sistema operativo ed altre amenità del genere. In un calcolatore, una qualsiasi procedura non è altro che un gruppo ben coordinato di istruzioni in linguaggio macchina.
In altre parole possiamo affermare che il Simon's Basic non è più potente del Basic del C/64 ma, unicamente, che dispone di un certo numero di routine richiamabili, invece che tramite Sys, per mezzo di parole codice, dette "istruzioni".

L'enciclopedia in linguaggio macchina che periodicamente pubblichiamo può benissimo rappresentare un ulteriore esempio di come un computer, opportunamente "caricato" da varie routine, svolga compiti che prima non era in grado di effettuare con semplicità.
Anche le routine della serie "Enciclopedia Basic" possono rappresentare, a tutti gli effetti, delle espansioni di comandi; analogamente un qualsiasi programma, come ad esempio un videogioco, può esser visto come un'estensione di comandi che, attivati da joystick, conducono all'elaborazione richiesta: generazione di musiche, animazioni, lancio di proiettili e così via.
I listati pubblicati nel presente inserto hanno uno scopo ben preciso, vale a dire quello di dimostrare che non è corretto affermare che una particolare versione di un linguaggio (il Basic, nel nostro caso) è più potente di un'altra, ma solo che è più versatile, oppure più semplice; quasi mai più veloce, e questo perchè le procedure da portare a termine devono necessariamente passare attraverso vie obbligate, percorsi ineludibili, confronti inevitabli.

## I listati

In alcune prove di velocità, riportate su alcune riviste, compaiono spesso rilevamenti sul tempo impiegato per completare un ciclo For...Next "a vuoto", come quello che segue:

```
100 Ti$="000000"
110 For X=1 to 10000
120 Next: Print Ti$
```

Prove di tal genere, tuttavia, tendono a falsare l'opinione che il lettore può trarre; capita raramente, infatti, di far compiere cicli a vuoto nel corso di un'elaborazione. Molto più spesso, infatti, si richiede ad un calcolatore di svolgere funzioni per un numero di volte determinato dal ciclo For...Next. Un computer, insomma, può risultare velocissimo svolgendo un For...Next a vuoto, ma lentissimo se conivolto in operazioni più sofisticate.

Il primo listato non ha, come già detto, un'applicazione pratica, ma servirà a far capire in quanti modi diversi è possibile far svolgere una stessa funzione ricorrendo anche a gruppi di istruzione poco noti.

Supponiamo di voler memorizzare, in una matrice bidimensionale capace di ospitare 1000x2 elementi, i mille numeri nella prima colonna e i corrispondenti valori della radice quadrata nell'altro.

In questo modo il computer è costretto ad attivare, per elaborare il pur semplice programma, numerose routine: quella relativa al ciclo For...Next in cui è richiesto, tra l'altro, un preciso Step; quella per la memorizzazione, all'interno di una matrice, di un valore intero (sotto forma, però, decimale); quella che elabora la radice quadrata; quella che incrementa l'indice X responsabile della corretta allocazione dei valori desiderati al posto giusto.
Le Rem in coda al programma rivelano tempi che sembrano paradossali: pur se la differenza del tempo riscontrato tra C/64

e C/16 può essere attribuita ad una differente taratura del quarzo, il tempo nettamente maggiore riscontrato sul C/128 fa nascere perplessità sulla presunta superiorità del C/128 rispetto ad un comune C/64.

Eppure i tempi sono proprio quelli riportati: il C/128 (in modo, ovviamente, 128) deve impiegare un tempo nettamente maggiore perchè le variabili sono allocate in un banco, il programma in un altro banco e l'interprete in un altro banco ancora! Il microprocessore, insomma, deve necessariamente selezionare più di un banco a seconda se è il momento di interpretare un'istruzione, rintracciare una variabile o eseguire una routine in linguaggio macchina che esegua un particolare compito; in tutto questo trambusto il tempo passa e, alla fine, risulta maggiore di quello impiegato da un C/64 che non deve saltare da un banco all'altro dal momento che l'intero programma viene elaborato nell'unico banco standard messo a diposizione dalla macchina.

Il secondo listato assolve allo stesso compito del precedente, ma ricorrendo ad istruzioni diverse; un ciclo If...Then è incaricato di eseguire 1000 volte il ciclo che, nel programma precedente, era affidato al For...Next. Si noti che il numero di istruzioni è pressochè identico e che il tempo impiegato non subisce variazioni di rilievo.

Per effettuare il conteggio, il terzo listato ricorre ad una tecnica, poco nota agli utilizzatori del Basic, basata sul confronto tra due variabili.

Se, ad esempio, scrivete...

A=100

...e subito dopo...

Print A=100

...il risultato sarà -1 che indicherà, appunto, che la variabile A è eguale a 100.

Se, invece, digitate...

Print A=23

...otterrete, in risposta, zero che segnalerà la mancata corrispondenza tra il valore 23 (preso a caso) e il contenuto di A.

Oltre che al segno eguale, è possibile ricorrere agli altri due simboli del confronto, cioè maggiore (>) e minore (<). Anche in questo caso se il confronto risulterà, in generale, vero, il risultato varrà -1; zero, invece, se falso.

Il comando On Var...Goto dovrebbe, invece, esser noto ai lettori: a seconda del valore della variabile Var, infatti, il programma proseguirà ad una delle linee Basic indicate nello stesso comando.

Consideriamo, ad esempio, il seguente segmento di programma:

```
100 On A Goto 200,210,220
110 Rem riga successiva
```

Arrivata a questo punto del programma, l'elaborazione proseguirà alla linea successiva 110 se "A" risulterà minore del valore unitario oppure maggiore o eguale a 4; alla riga 200 se il valore sarà maggiore di 1 e minore di 2; alla riga 210 se compreso tra 2 e 3; alla 220 se minore di 4.

Sfruttando le due tecniche appena descritte è possibile far compiere al calcolatore i mille cicli effettuati nei casi precedenti, con un incremento del tempo, tutto sommato, piuttosto contenuto.

E' quindi probabile che l'interprete Basic, nei tre casi finora esaminati, pur se, in apparenza, svolge lo stesso compito in modo totalmente diverso, è costretto, a livello di linguaggio macchina, ad eseguire le stesse routine che sono interpellate sia nel ciclo For...Next, sia nell'If...Then, sia nell'On...Goto.

E' più che probabile, insomma, che il Basic effettui la stessa verifica di fine conteggio nei tre casi del Next, del Then, dell'On.

Con ciò abbiamo dimostrato che, molto spesso, l'elaborazione di un programma, pur avendo forma diversa, segue percorsi piuttosto simili tra loro.

## Un altro mito crolla

Gli altri programmi non possono far altro che dimostrare, purtroppo, che il C/128, almeno a livello Basic, perde su tutto il

fronte a causa della complessa gestione dei banchi di memoria.
Non possiamo fare a meno di rilevare che la maggior disponibilità di comandi Basic si traduce in un incremento del tempo di elaborazione.

Spicca, invece, la notevole velocità degli stessi programmi che, con minime modifiche, girano allo stesso modo su Amiga.

Addirittura il comando Fill, che richiede quasi due minuti con il C/128, è talmente veloce su Amiga che è stato necessario farlo eseguire 100 volte per ottenere una rilevazione di tempi accettabile (1 secondo per 25 cicli!). E' ovvio che Amiga, o meglio il suo interprete Basic, non ricorre alla tecnica del tracciamento di tante linee finchè non riempie la figura; con tutta probabilità, nel caso di comandi Fill, viene attivato il coprocessore grafico che si incarica di svolgere il compito richiesto.

Tutto ciò è in sintonia con quanto affermato in precedenza: per svolgere con successo complesse procedure è indispensabile ricorrere a processori, e sistemi in generale, ben più potenti e veloci di un semplice, pur se glorioso, otto bit.
Gli ultimi due programmi hanno il compito di dimostrare, se mai ve ne fosse bisogno, che oltre alla velocità del micro deve esser considerata la versatilità del sistema provato.

Il primo programma (estratto da un listato scritto dal nostro collaboratore Roberto Ferro), che gira su C/16, Plus 4 e C/128 (su quest'ultimo, però, gira a velocità ridotta...) consente, servendosi di un joystick, di disegnare sulla pagina video in alta risoluzione. Chi avrà modo di provarlo non potrà fare a meno di notare l'estrema lentezza di gestione della penna.

Le stesse funzioni possono esser svolte, su Amiga, dal minimicro programma pubblicato da usare servendosi del mouse: la velocità di operazione, in questo caso, è semplicemente spaventosa, anche se paragonata con quella ottenibile con il C/16 caricando il programma di prima.
Ciò dovrebbe bastare a convincere il lettore che un sistema può esser definito "superiore" solo se, nei casi più comuni, riesce ad eseguire compiti che, assegnati a computer "inferiori" richiedano tempi operativi, versatilità di stesura e facilità di controllo migliori sotto tutti i punti di vista.

**Listato 1**

Un esperimento semplicissimo: allocazione, in una matrice bidimensionale, di 1000 numeri e dei corrispondenti valori delle radici quadrate. Il tempo richiesto per l'intera elaborazione non dovrebbe lasciare dubbi sulle reali velocità dei quattro computer considerati.

```
100 REM LISTATO N.1
110 REM PROVA DI VELOCITA' FOR...NEXT
120 :
130 DIM A(1000,1): REM CREAZIONE SPAZIO
140 INPUT"DA/A/STEP";A,B,C
150 :
160 TI$="000000":REM AZZERAMENTO TEMPO
170 :
180 FOR I=A TO B STEP C
190 A(X,0)=X:A(X,1)=SQR(X):X=X+1
200 NEXT
210 :
220 PRINT TI$:STOP:REM VERIFICA TEMPO
230 :
240 REM ESAME MATRICE
250 FOR Y=0 TO X-1
260 PRINT"VALORE"A(Y,0);"RADICE"A(Y,1)
270 NEXT
280 REM RISULTATI (DA 1 A 100 STEP .1)
290 REM COMMODORE  64: 107 SECONDI
300 REM COMMODORE 128: 116 SECONDI
310 REM COMMODORE  16: 100 SECONDI
320 REM AMIGA     500:   5 SECONDI
```

**Listato 2**

La stessa funzione del primo listato può essere svolta da un ciclo If...Then che si occupa, stavolta, di effettuare salti, o meno, a seconda del valore raggiunto dalla variabile Y. I tempi rilevati, molto simili a quelli precedenti, dimostrano che la "forma" non influisce in maniera evidente sul nocciolo dell'elaborazione.

```
100 REM LISTATO N.2
110 REM PROVA DI VELOCITA' IF...THEN
120 :
130 DIM A(1000,1): REM CREAZIONE SPAZIO
140 INPUT"DA/A/STEP";A,B,C:Y=A
150 :
160 TI$="000000":REM AZZERAMENTO TEMPO
170 :
180 IF Y<B THEN 200
190 GOTO230
200 A(X,0)=Y:A(X,1)=SQR(Y)
210 X=X+1:Y=Y+C:GOTO 180
220 :
230 PRINT TI$:STOP:REM VERIFICA TEMPO
240 :
250 REM ESAME MATRICE
260 FOR Y=0 TO X-1
270 PRINT"VALORE"A(Y,0);"RADICE"A(Y,1)
280 NEXT
290 REM RISULTATI (DA 1 A 100 STEP 0.1)
300 REM COMMODORE  64: 114 SECONDI
310 REM COMMODORE 128: 126 SECONDI
320 REM COMMODORE  16: 108 SECONDI
330 REM AMIGA     500:   5 SECONDI
```

**Listato 3**

Anche il ricorso al comando On...Goto, e alla determinazione di un valore mediante confronti, non aumenta in modo considerevole il tempo di elaborazione, che rimane circoscritto nell'ambito del 10%

```
100 REM LISTATO N.3
110 REM VERIFICA ON...X...GOTO
120 :
130 DIM A(1000,1): REM CREAZIONE SPAZIO
140 INPUT"DA/A/STEP";A,B,C:X=A
150 :
160 TI$="000000":REM AZZERAMENTO TEMPO
170 :
180 REM CONDIZIONE (DA...A...STEP)
190 ON 2+(X<B) GOTO 220,260
200 END
210 REM FUNZIONE DA SVOLGERE
220 A(I,0)=X:A(I,1)=SQR(X):I=I+1
230 :
240 X=X+C:GOTO 170:REM INCREMENTO
250 :
260 PRINT"FINE":REM FINE CICLO
270 PRINTTI$:STOP:REM VERIFICA TEMPO
280 :
290 REM ESAME MATRICE
300 FOR Y=0 TO I-1
310 PRINT"VALORE"A(Y,0);"RADICE"A(Y,1)
320 NEXT
330 REM RISULTATI (DA 1 A 100 STEP .1)
340 REM COMMODORE  64: 118 SECONDI
350 REM C/64+GW-BASIC: 120 SECONDI
360 REM COMMODORE 128: 132 SECONDI
370 REM COMMODORE  16: 113 SECONDI
380 REM      AMIGA 500:   6 SECONDI
```

**Listato 4**

Il comando On...Goto viene utilizzato, in questo listato, due volte: la prima per effettuare il conteggio; la seconda per evitare l'illegal quantity error che potrebbe verificarsi tentando di calcolare il logaritmo di un numero negativo.

```
100 REM LISTATO N.4
110 REM ON...GOTO CON VERIFICA ERRORE
120 :
130 INPUT"DA/A/STEP";A,B,C:X=A
140 :
150 REM CONDIZIONE (DA...A...STEP)
160 ON 2+(X<B) GOTO 190,280
170 :
180 REM CONTROLLO ERRORE
190 ON 2+(X<0) GOTO 220,250
200 :
210 REM MESSAGGIO ERRORE
220 PRINT"ERRORE:X=";X;"<0"
230 GOTO 260
240 :
250 PRINT"VALORE"X;"LOGARITMO"LOG(X)
260 X=X+C:GOTO 160:REM INCREMENTO
270 :
280 PRINT"FINE":REM FINE CICLO
```

---

**Listato 5**

Il ciclo Do...Until è disponibile soltanto sui computer C/16 e C/128. Il tempo di elaborazione rimane in linea con quelli visti in precedenza.

```
100 REM LISTATO N.5
110 REM PROVA DI VELOCITA' DO...UNTIL
120 :
130 DIM A(1000,1): REM CREAZIONE SPAZIO
140 INPUT"DA/A/STEP";A,B,C:Y=A
150 :
```

```
160 TI$="000000":REM AZZERAMENTO TEMPO
170 :
180 DO UNTIL Y>B: REM PRINT Y
190 A(X,0)=Y:A(X,1)=SQR(Y)
200 X=X+1:Y=Y+C
210 LOOP
220 :
230 PRINT TI$: REM VERIFICA TEMPO
240 :
250 REM RISULTATI (DA 1 A 100 STEP .1)
260 REM COMMODORE 128: 122 SECONDI
270 REM COMMODORE  16: 106 SECONDI
```

**Listato 6**

Il ciclo Do...While (con sintassi leggermente diversa nel caso dell'Amiga) rappresenta un nuovo modo per effettuare il conteggio che, nel modo più banale, potrebbe esser risolto con un For...Next.

```
100 REM LISTATO N.6
110 REM PROVA DI VELOCITA' DO...WHILE
120 :
130 DIM A(1000,1): REM CREAZIONE SPAZIO
140 INPUT"DA/A/STEP";A,B,C:Y=A
150 :
160 TI$="000000":REM AZZERAMENTO TEMPO
170 :
180 DO WHILE Y<B:REM PRINT Y;
190 A(X,0)=Y:A(X,1)=SQR(Y)
200 X=X+1:Y=Y+C
210 LOOP
220 :
230 PRINT TI$: REM VERIFICA TEMPO
240 :
250 REM RISULTATI (DA 1 A 100 STEP .1)
260 :
270 REM COMMODORE 128: 123 SECONDI
280 REM COMMODORE  16: 106 SECONDI
290 REM AMIGA     500:   5 SECONDI
```

**Listato 7**

La forma sintattica If...Then...Else, disponibile su numerose versioni del Basic, consente anch'essa di effettuare un ciclo. Scegliere una forma sintattica oppure un'altra è solo una questione di... gusti o di opportunità in alcuni casi particolari.

```
100 REM LISTATO N.7
110 REM PROVA DI VELOCITA' IF/THEN/ELSE
120 :
130 DIM A(1000,1): REM CREAZIONE SPAZIO
140 INPUT"DA/A/STEP";A,B,C:Y=A
150 :
160 TI$="000000":REM AZZERAMENTO TEMPO
180 IF Y<B THEN 190:ELSE GOTO 220
190 A(X,0)=Y:A(X,1)=SQR(Y)
200 X=X+1:Y=Y+C:GOTO 180
210 :
220 PRINT TI$:REM VERIFICA TEMPO
240 REM RISULTATI (DA 1 A 100 STEP .1)
250 REM C/64+GW-BASIC: 115 SECONDI
260 REM COMMODORE 128: 126 SECONDI
270 REM COMMODORE  16: 108 SECONDI
280 REM AMIGA     500:   6 SECONDI
-----------------------
```

**Listato 8**

Il comando Begin...Bend, che consente di svolgere numerose elaborazioni nel caso si verifichi un particolare evento logico, è disponibile solo nel C/128. Si noti come in questo caso, rinunciando alla visualizzazione dello schermo, il tempo richiesto per l'intero ciclo diminuisca notevolmente se si attiva il comando Fast disponibile, appunto, sul C/128.

```
100 REM LISTATO N.8
110 REM TEMPO BEGIN...BEND PER C/128
120 :
130 DIM A(1000,1): REM CREAZIONE SPAZIO
140 INPUT"DA/A/STEP";A,B,C:Y=A
150 :
160 TI$="000000":REM AZZERAMENTO TEMPO
```

```
170 REM FAST
180 IF Y<B THEN BEGIN: REM PRINT Y;
190 A(X,0)=Y:A(X,1)=SQR(Y)
200 X=X+1:Y=Y+C
210 BEND: GOTO 180
220 REM SLOW
230 PRINT TI$: REM VERIFICA TEMPO
240 :
250 REM RISULTATI (DA 1 A 100 STEP .1)
260 :
270 REM C/128 (SLOW): 126 SECONDI
280 REM C/128 (FAST):  41 SECONDI
-----------------------------------
```

**Listati 9, 10 e 11**

Anche ricorrendo al comando Fast, il C/128 richiede un tempo notevolmente maggiore rispetto a quello richiesto dall'Amiga.

```
100 REM LISTATO N.9
110 REM PROVA DI VELOCITA'
120 REM DISEGNO DI CIRCONFERENZE
130 :
140 GRAPHIC 1,1: TI$="000000"
150 REM FAST
160 FOR I=1 TO 10 STEP.2
170 CIRCLE 1,100,100, I*5
180 NEXT: PRINT TI$
190 REM SLOW
200 GET A$:IF A$="" THEN 200
210 GRAPHIC 0,0
220 REM C-16   TEMPO: 43 SECONDI
230 REM C-128 (SLOW): 53 SECONDI
240 REM C-128 (FAST): 25 SECONDI
250 REM AMIGA    500:  3 SECONDI
```

□ □ □

```
100 REM LISTATO N.10
110 REM PROVA DI VELOCITA'
120 REM TRACCIAMENTO RETTANGOLI
130 :
```

```
140 GRAPHIC 1,1:TI$="000000"
150 FOR I=1 TO 10 STEP.05
160 BOX 1,100,100, I*10, I*20
170 NEXT:PRINT TI$: GRAPHIC 0,0
180 REM C-16  TEMPO: 19 SECONDI
190 REM C-128 TEMPO: 20 SECONDI
200 REM AMIGA TEMPO:  1 SECONDO
```

□ □ □

```
100 REM LISTATO N.11
110 REM PROVA DI VELOCITA'
120 REM TRACCIAMENTO LINEE INCLINATE
130 :
140 GRAPHIC 1,1:TI$="000000"
150 FOR I=1 TO 10 STEP.05
160 DRAW 1,100,100 TO I*5 ,I*20
170 NEXT:PRINT TI$:GRAPHIC 0,0
180 REM C-16/128 TEMPO: 10 SECONDI
190 REM AMIGA    TEMPO: 1 . SECONDO
---------------------------
```

**Listato 12**

Il caso emblematico del comando Fill che, nel caso dell'Amiga (pur se con sintassi particolare), è incredibilmente veloce e, comunque, nettamente superiore a quelli prima osservati. E' evidente che il potente computer affida al co-processore grafico il difficile compito di colorare un'area delimitata.

```
100 REM LISTATO N.12
110 REM PROVA DI VELOCITA'
120 REM RIEMPIMENTO DI UN RETTANGOLO
130 :
140 GRAPHIC1,1
150 FAST
160 DRAW 1,1,1 TO 310,1 TO 310,190
170 DRAW TO 1,190 TO 1,1
180 TI$="000000":PAINT 1,100,100
190 PRINT TI$
200 SLOW
210 GET A$:IF A$="" THEN 210
```

```
220 GRAPHIC0,0
230 REM C-16  TEMPO: 103 SECONDI
240 REM C-128  SLOW: 109 SECONDI
250 REM C-128  FAST:  33 SECONDI
260 REM AMIGA (100 VOLTE) 4 SECONDI
```

```
REM riempimento
REM di area rettangolare
REM con Amiga 500
ti$= TIME$
FOR i=1 TO 100
LINE(1,1)-(310,190),,bf
CLS: NEXT
PRINT ti$:PRINT TIME$
REM 4 secondi
```

---

**Listato 13**

Forse non è possibile scrivere un listato più breve, per C/16 e C/128, che consenta, con il joystick, di disegnare in alta risoluzione. Qui mancano i controlli che evitino l'elaborazione di quantità illegali. Volendo prevedere, ed evitare, questi casi sarebbe necessario aggiungere altri comandi ed istruzioni che parteciperebbero a rallentare l'elaborazione del programma stesso.

```
100 REM DISEGNI IN HI-RES PER C-128,
110 REM PER C/16 E PLUS-4 (+JOYSTICK)
120 REM FIRE ABILITA/DISAB. DISEGNO
125 :
130 GRAPHIC1,1:A=160:B=100
140 J=JOY(1)
150 ONJGOTO170,180,190,200,210,220,230,
    240
160 IF J>127 THEN 310:ELSE 140
170 X=-1:GOTO 250
180 X=-1:Y=1:GOTO 250
190 Y=1:GOTO 250
200 Y=1:X=1:GOTO 250
210 X=1:GOTO 250
220 X=1:Y=-1:GOTO 250
```

```
230 Y=-1:GOTO 250
240 Y=-1:X=-1
250 DRAW C,A,B:A=A+Y:B=B+X
260 LOCATE A.B:C=RDOT(2)
270 IF C=1THEND=0:ELSED=1
280 IF K=3THENGOSUB330
290 DRAW D,A,B
300 X=0:Y=0:GOTO140
310 GOSUB340:IFK=3THENK=0:C=1:ELSEK=3
320 GOTO140
330 DRAW 1,A-Y,B-X:RETURN
340 DO WHILE JOY(1)>127:LOOP:RETURN
350 END
- - - - - - - - - - - - - - - - - - - - - - - - - -
```

**Listato 14**

Ecco, per Amiga, una versione minima di Koala: il mouse si rivela uno strumento incredibilmente veloce ed affidabile per la gestione del listato che non presenta alcuna difficoltà di comprensione. La velocità con cui si può disegnare è notevole e non sono possibili "equivoci " dal momento che l'interprete dell'Amiga "conosce" benissimo, in qualsiasi momento, le coordinate del mouse.

```
REM minimicro koala
REM per Amiga
loop:
x=MOUSE(0):a$=""
IF x=0 THEN loop
PSET(MOUSE(1),MOUSE(2)),1
GOTO loop
REM per disegnare tener
REM premuto il tasto sin.
```

---

```
REM mini koala
REM in quattro colori
REM per Amiga
a=1
loop:
x=MOUSE(0):a$=""
```

```
a$=INKEY$:
IF LEN(a$)>0 THEN a=a+1:IF a>3 THEN a=0
IF x=0 THEN loop
PSET(MOUSE(1),MOUSE(2)),a
GOTO loop
REM premendo un tasto della
REM tastiera il colore del tratto
REM cambia fino al quarto posibile
REM (uno dei tre e' eguale al fondo)
```